ToolBox Mag

n° 4

Avril - Mai 1991

Prix: 590F par abonnement.

Sommaire de la revue :

Infos: **Sommaire de la disquette n° 4**	ToolBox Mag	Page 2
Editorial: **Fort et vert ...**	E. Weyland	Page 3
Infos: **Le système nouveau est arrivé**	J.Y. Bourdin	Page 4
Dossier trilingue : décorez votre bureau		
Asm: **La communication entre logiciels**	P. Desnoues	Page 5
C: **C facile n° 4, des inits en C**	F. Uhrich	Page 10
Pas: **Ecrire des Inits en Pascal**	B. Fournier	Page 13
News: **Exotica**	F. Hermellin	Page 14
Programme: **Mac Eraser, le source**	O. Goguel	Page 16
Programme: **Et voici le GS Roms 05**	P. Desnoues	Page 19
Essai logiciel: **InWords, le GS sait lire**	B. Fournier	Page 20
Programme: **Border Scroll**	D. Delay	Page 22
Initiation: **Le GS pour débutants, n°1**	E. Weyland	Page 24
Infos: **L'égalité Mac-GS, enfin !**	ToolBox Mag	Page 28
Bidouille: **Le monde à l'endroit**	J.Y. Bourdin	Page 29
Son: **MidiSynth Player**	J.P. Charpentier	Page 30
Infos: **Le plus grand réseau du monde**	P. Pointu	Page 32
News du terroir: **Made in France**	B. Fournier	Page 33
Jeu: **Fun Beyond GS**	O. Goguel	Page 34
News: **GS News**	E. Weyland	Page 36
Programme: **GE 80**	R. Mange	Page 40
Disque dur: **Les mégas sans la galère**	J.Y. Bourdin	Page 44
HyperMédia: **Premiers conseils HyperCard**	J.Y. Bourdin	Page 47
Langages: **De TML à Orca-Pascal**	M. Racine	Page 47
Ressources: **N° 3, se modifier soi-même**	B. Fournier	Page 48
Dialogues: **Courrier des lecteurs**	Vous	Page 50

Présentation de la disquette ToolBox-Mag n° 4

Nous ne nous lasserons jamais de le répéter, ToolBox Mag se lit GS en route: **lisez la disquette!** Voici son catalogue:

/ANNONCES: nos petites annonces gratuites (pour les abonnés...)

/BABAR: l'application à lancer *MidiSynth.Play* de Jean-Pierre Charpentier, et ses sources. Explications pages 30-31.

/BORDER.SCROLL: le programme de Dominique Delay permettant d'agrandir l'écran de votre GS! Explications pages 22-23.

/C.FACILE: 4e édition. F. Uhrich nous montre comment écrire des Inits en Orca/C et APW/C. "C quoi, un Init?" Voyez pages 10 à 12.

/CRUNCH: contient les fichiers de l'animation Graphique-Son de présentation de la disquette. *Bootez la disquette,* c'est du D. Delay.

/DERNIERE.HEURE: nous avons réussi à garder quelques blocs sur le disque pour des infos ou corrections éventuelles de dernière minute.

/DIVERS: ce catalogue fourre-tout contient:

- *Bram.Util*, application de J.L. Darbon. Voyez page 26.
- *DefaultFile*, fichier pour Genesys de J.Y Bourdin. Voir page 31.
- *Franciser.5.04*, fichier texte à lire où Yvan Kœnig explique comment il a francisé le système GS/OS 5.04. Voyez page 4.
- *Ote.Finder.Data*, un petit programme Applesoft du même Yvan Kœnig pour nettoyer les pollutions déposées par le Finder sur nos disques.
- *Lancez.Moi* et *GetVersion(.S)*: un mini-programme et son source de J.Y. Bourdin. Voyez page 18. *S'il vous dit que vous n'êtes pas à jour, c'est que vous ne l'êtes pas*: voyez page 4.

/FUN.BEYOND.GS: le super-jeu en six versions d'Olivier Goguel et du FTA. Plus le source, pour que vous fassiez la septième! Voir page 34.

/GE.80: le magnifique progiciel éditeur d'écrans 80 colonnes de Régis Mange. Eh oui, tout en P8, et super-pro. Sans compter tout ce qu'il y a à repiquer dans les sources! Voyez pages 40 à 43.

/ICONS: icônes.

/JOLI.BUREAU.INIT: tout le nécessaire pour décorer votre bureau, en deux versions, de Patrick Desnoues. Voyez pages 5 à... 13!

/MAC.ERASER: ou 524 octets pour le prix de 512, par Olivier Goguel. Voyez pages 16 à 18.

/MACROS.APW: ou le 65832 par soft de Patrick Desnoues. Toutes explications dans le fichier *Lisez.Moi* d'accompagnement.

/PASCAL.INIT: vous l'aviez en C et en assembleur, le voici en Pascal, par Bernard Fournier. Voyez page 13. Si après ça, vous gardez votre vieux bureau bleu, autant passer sur IBM!

/RADIO.AMATEURS: trois fichiers texte à lire de Patrick Pointu. Quel rapport avec le GS? Voyez page 32.

/RAMTOOLS.INIT: ou le GS Roms 05 selon Patrick Desnoues. Voyez page 19.

/RESSOURCES: 3e édition. Comment se modifier soi-même sans se mordre la queue, selon Bernard Fournier. Voir pages 48-49.

/TOOL219.V.1.1.1: dernière version du célèbre Tool de ToolBox Mag, par Olivier Goguel, avec un fichier de modifications du source. Mais aussi un catalogue **/ORCA.PASCAL**, contenant les interfaces pour Orca-Pascal et des versions Orca des programmes de démonstration, par Michel Racine. Toutes explications page 47.

Finder.Root est un fichier pour le Finder, *PRODOS* est un faux.

ToolBox-Mag

Directeur de la Publication
Eric Weyland

Rédacteur en Chef
Jean-Yves Bourdin

Mise en pages
Dominique Digoy

Secrétariat
Janine Loiseleux

Conseil de Rédaction
Jean-Pierre Charpentier
Patrick Desnoues
Bernard Fournier
Olivier Goguel
Stéphan Hadinger
François Hermellin
Yvan Kœnig
Hubert Loiseleux
Claude Pélisson
Jean-Luc Schmitt
François Uhrich

Rédaction, Siège Social
ToolBox-Mag
6, rue Henri-Barbusse
95100 Argenteuil - France
Téléphone: (1) 30 76 18 64
Télécopie: (1) 39 47 44 08

Impression
Imprimerie /Reprographie /Graphisme - Argenteuil

Les auteurs de ToolBox-Mag n° 4

Olivier Bailly-Maître
Jean-Yves Bourdin
Jean-Pierre Charpentier
Jean-Luc Darbon
Domique Delay
Patrick Desnoues
Bernard Fournier
Olivier Goguel
François Hermellin
Yvan Kœnig
Régis Mange
Sébastien Mousseron
Patrick Pointu
Michel Racine
François Uhrich
Eric Weyland

Avertissements légaux

ToolBox-Mag est un magazine qui prend l'Apple II GS au sérieux. De ce fait même, il est indépendant d'Apple Computer et d'Apple Computer France, auxquels il n'est, grâce au ciel, pas rattaché.
ToolBox-Mag est disponible uniquement par abonnement et par correspondance. Le magazine est composé inséparablement d'une revue sur papier et d'une disquette magnétique pour ordinateur Apple // GS. Certains avertissements légaux, modes d'emploi, etc, ou tous autres textes, peuvent être placés sur la disquette, et nécessiter un Apple II GS pour être lus: ils n'en sont pas moins écrits dans ToolBox-Mag, dont ils font partie intégrante.

Tout le contenu de la disquette et de la revue ToolBox-Mag sont entièrement sous Copyright de la Société ToolBox, et sont déposés à l'Agence pour la Protection des Programmes. Tous droits de traduction et de reproduction intégralement réservés pour tous pays.
Aucune reproduction, même partielle, et sous quelque forme que ce soit, de la disquette comme de la revue ToolBox-Mag, ou d'un des programmes qu'elles contiennent, ne pourra avoir lieu sans accord préalable et écrit de ToolBox-Mag. Copier ToolBox-Mag, c'est illégal, et cela vise à torpiller le seul magazine français sur l'Apple // GS: nous serons alors sans pitié.

"GS" est un sigle déposé par les automobiles Citroën. La Pomme est un symbole déposé par Eve et le Serpent. Apple, Apple II GS, le logo Apple, Macintosh et le logo Macintosh, ImageWriter, LaserWriter et 'Fatal System Error 911' sont des marques déposées d'Apple Computer. AppleWorks-GS est une marque déposée par Claris-USA, à l'insu de Claris-France. ToolBox™ et ToolBox-Mag™ sont des marques déposées de la Société ToolBox®...

ToolBox-Mag ne peut pas offrir une garantie supérieure à celle qu'offre le chapitre "Limitations de Garantie et de Responsabilité" des documentations officielles Apple, soit quasiment rien. Ecrit par des amateurs du GS, il s'engage néanmoins à publier des programmes ne contenant pas plus de bugs que GS Write, et à maintenir dans ses colonnes un standard de compétence au moins égal au niveau moyen des concessionnaires Apple français en matière de GS...

EDITORIAL

Fort et vert...

Rafale de nouveautés, en ce début de printemps, sur notre machine préférée:
• Un nouveau système, **GS/OS 5.04**, cette fois-ci disponible simultanément en France et aux USA, grâce à l'un de nos auteurs, Yvan Kœnig.
• Le magnifique outil **MidiSynth** et son logiciel **SynthLab** sont désormais en vente libre: vous ne saviez pas encore vraiment ce que signifie le "S" de "GS". Bien entendu, Jean-Pierre Charpentier commence dès ce numéro à nous montrer comment on s'en sert.
• **HyperCard GS** est désormais dans le commerce: c'est bien, comme le dit A2 Central, l'Applesoft des années 90, la programmation en mode Graphique/Son à la portée de tous.
• **InWords**, de WestCode Software, lui aussi commercialement disponible, apprend à lire à notre GS (voyez l'étude de Bernard Fournier).
• **Genesys 1.2**, de SSSI, enfin, qui écrit à votre place une application complète avec tous ses menus, toutes fenêtres, tous ses boutons à cocher, ses boîtes de dialogue, etc: c'est la programmation en mode "desktop" à la portée de tous.

Ce numéro 4 est aussi la preuve vivante d'une des grandes caractéristiques du GS: **l'inventivité des programmeurs**, qui découle de leur **liberté**. Jouez au "Fun Beyond GS" du FTA, utilisez le magnifique GE 80 de Régis Mange (en P8!), écrivez dans la bordure de votre écran, mettez tous vos outils en mémoire, dessinez-vous le fond de bureau que vous souhaitez, et bien d'autres choses encore: tout cela dans ce numéro 4.
Cette inventivité a une source: la machine elle-même. "Merci, Apple", écrit Patrick Desnoues en fêtant son entrée à notre Conseil de Rédaction. Même quand on en fait un usage imprévu, c'est bien encore elle qui le suscite.

Reconnaissons-le, **tous ces progrès ont un prix**: l'usage du GS n'est pas ce qu'il y a de plus facile pour le débutant, ni même pour l'habitué, qui doit souvent changer d'habitudes. Notre magazine essaie d'**aider chacun à ces changements**.
Par exemple, chacun de nos programmes comprend une partie "pour l'utilisateur": nous avons donc supprimé les , chaque article étant à plusieurs niveaux. Nous continuons, bien sûr, les nouvelles du monde, de France et... du monde extérieur. Nous publions patches et conseils d'utilisation. Nous démarrons aussi, dans ce numéro même, une série pour les débutants.

La prochaine fois qu'on vous ressortira l'inusable fable de la "mort de l'Apple II", sortez votre arme: le Bulletin d'abonnement à ToolBox Mag !

ERIC WEYLAND

GS/OS 5.04 : le système nouveau est arrivé !

J.Y. Bourdin

Grande nouvelle: la dernière version du système de l'Apple II GS, **GS/OS 5.04, est disponible en France, en version française, en même temps qu'aux USA.**

Quand on sait qu'il faut des mois et des mois à la bureaucratie Apple pour réaliser une version française du système, on devine ce qui se passe: cette bureaucratie a été court-circuitée, et quelqu'un a réalisé une version à coups de désassembleur. Devinez qui? Yvan Kœnig, bien sûr!
Vous trouverez sur la disquette ToolBox Mag une note "technique" de notre éminent collaborateur expliquant ses choix de francisation. Pour ma part, je me contenterai de vous expliquer pourquoi vous ne pourrez pas vous en passer.

Car cette nouvelle version du système est **indispensable** à tout utilisateur d'Apple II GS:
• Quantité de bugs, dont certains vraiment gênants (dans TextEdit et le Resource Manager par exemple), ont disparu.
• Le nouveau driver d'ImageWriter permet une impression de listings en mode rapide tout en restant en mode graphique: cela signifie que **vous ne verrez plus dans ToolBox Mag ces listings TML Pascal sans accents qu'on trouvait dans les premiers numéros**. On peut écrire ses commentaires normalement sous TML, et on imprime les fichiers avec Appleworks-GS ou tout autre traitement de textes GS.
• **La francisation du clavier est automatiquement celle d'Yvan Kœnig**: c'est désormais la colonne de droite du tableau pages 18/19 de ToolBox Mag 3 que vous avez par défaut dans votre système. J'ai déjà dit la supériorité de la formule Kœnig sur l'ancienne formule Apple.
• Si vous n'avez pas la dernière version du système, **vous serez incapable de faire tourner les programmes des disquettes ToolBox Mag**, (comme nombre d'entre vous s'en sont déjà aperçus!). Il est absolument exclu que ToolBox Mag demande à ses auteurs de régresser dans leur usage du GS: **ToolBox Mag est incompatible avec les systèmes GS périmés.**
• **Vous ne pourrez pas faire tourner non plus Platinum Paint et HyperCard**, qui sont des nouveautés logicielles majeures du GS. Bref, le 5.04 est le système d'aujourd'hui.

Attention, la bureaucratie Apple étant sur la touche, **la diffusion du 5.04 est un peu spéciale**: vous risquez de ne guère le trouver pour le moment chez les concessionnaires Apple ordinaires, mais seulement chez les commerçants qui soutiennent réellement le GS: la liste est courte, et vous la connaissez.
Il ne contiendra aucune documentation spécifique sur papier (il n'en a pas besoin). Vous aurez deux formules pour l'acquérir:
• si vous n'avez pas le système 5.02 français (honte à vous), vous paierez simplement 100 F plus cher le système complet. Vous aurez la documentation du 5.02, mais les disquettes seront celles du 5.04. Vous serez donc enfin à jour.
• si vous avez déjà le 5.02 français, il suffira de payer 100F (plus port) pour une disquette de mise à jour. Voici dans ce cas la procédure à suivre: faîtes une copie de vos deux disquettes du 5.02. Lancez votre 5.02 actuel, et passez sous le Finder. Introduisez la disquette "Mise à jour 5.04", et lancez le programme "Installation".
Ce programme se chargera de mettre à jour vos copies des disquettes système, il suffira de lui obéir. A la fin, vos deux disquettes 5.02 seront transmuées en système 5.04. Lancez alors le programme Installation de ces nouvelles disquettes système, pour installer le 5.04 sur le disque de votre choix: la documentation du 5.02 vous dit ce qu'il faut savoir à ce sujet.

C'est un peu compliqué, mais cela vous permet de payer le minimum, et d'avoir le nouveau système en même temps que les Américains.

Une précision enfin: ce système a des bugs, bien sûr, certains sont même déjà officiels. Mais la probabilité pour qu'un de ces bugs ait été rajouté à la francisation est **infinitésimale**. Yvan Kœnig n'est pas un des auteurs du 5.04: lisez son texte sur la disquette avant de lui transmettre vos "bug reports".

La communication entre logiciels, ou comment décorer son bureau

Patrick Desnoues

Si vous communiquez un peu sur Minitel, vous savez que, quand vous vous connectez sur certains serveurs, on commence par vous avertir que vous avez du courrier dans votre Boîte aux Lettres (BAL). De la même façon, vous pouvez laisser un message à quelqu'un: dès qu'il se connectera, il sera prévenu.
Eh bien, il en est de même pour les différentes applications du GS: depuis le ToolLocator v2.1, il existe **une BAL interne au GS**, appelée MessageCenter, et quand un programme démarre, il peut commencer par consulter son courrier.
Avec le système 5.0, arrive un nouveau type de message. Gageons qu'Apple ne s'arrêtera pas là. Sur Mac, ce type d'interface est largement utilisé, il faut que cela le devienne également sur GS, l'intégration des applications différentes étant réellement un gain pour l'utilisateur.
Appleworks-GS et quelques autres le permettent déjà: alors programmeurs, communiquez, d'autant plus que c'est très simple.

Vous connaissez certainement un **premier type de message**, qui existe depuis un moment: quand on clique sur l'icône d'un fichier d'AppleworksGS dans le Finder, on peut lancer l'application par les commandes 'Ouvrir' où 'Imprimer' du menu 'Fichier'.
Une fois l'application lancée , celle-ci teste si elle a du 'courrier'. L'outil MessageCenter (Miscellaneous Tool) va se charger de l'informer :
a) Est-ce qu'il y a un message ?
b) Si oui, le nom du fichier à traiter est donné.
c) Que fait-on de ce fichier (ouvrir, imprimer) ?
Voir l'exemple 1 qui traite de la réception d'un message. Pour envoyer un message vers une autre application, il suffit de communiquer le nom du fichier à traiter, le type de traitement, et de suivre l'exemple 2...

Mais **un nouveau type de message** est apparu avec le système 5.0. Cela m'a donné l'idée d'écrire deux petits Inits utilisant ces nouvelles propriétés (*Joli.Ecran* et *Joli.Motif*).

Ce type de message est d'ordre graphique: à chaque initialisation du Window Manager, celui-ci va aller voir s'il y a des informations pour lui. Si oui, **au lieu de l'écran bleu standard du bureau, il va afficher une image ou un motif de notre choix.**

Les sources complets des deux inits sont donnés dans les deux exemples. Exemple 3: Message de type Pattern (motif). Exemple 4: Message de type Picture (dessin).

Utilisateurs, copiez **dans cet ordre** à la fin de votre dossier */SYSTEM/SYSTEM.SETUP/ les trois fichiers suivants de /TOOLBOX.MAG.04/JOLIBUREAU.INIT/: 1/ JOLI.MOTIF; 2/ JOLI.ECRAN; 3/ ECRAN.BUREAU. Désactivez ensuite (avec le Finder) celui des deux fichiers inits que vous ne désirez pas voir actif. Puis redémarrez.

Merci Apple, on attend d'autres messages...

PS : ces exemples utilisent les librairies e16.Locator et e16.Window d'APW/Orca. Les exemples 1 et 2 ne sont pas directement réassemblables, ce sont des schémas pour vos propres programmes. A vous de les remplir avec vos messages. Pour réassembler les exemples 3 et 4, veillez à avoir mis le fichier /MACROS.APW/M16.A de la disquette ToolBox Mag 4 en tête de vos fichiers M16.X.

Trois fois le même !

Vous trouverez pages 8 et 9 de ce numéro **trois fois le même programme** (l'exemple 4 de Patrick Desnoues) permettant de charger le dessin de votre choix en fond de bureau.
Il s'agit de montrer comment **la même routine peut être écrite indifféremment en C, en Pascal ou en Assembleur.**
Du coup, vous apprendrez aussi comment écrire des Inits en C et Pascal, et comment y mélanger un peu d'assembleur (ce n'était pas si simple que ça en avait l'air...).

JYB

Exemple 1 : réception d'un message MessageCenter par une application

```
MessageCenter ENTRY
{Longueur maximum : 4 (MessNext) + 2 (MessType) + 2 (Info)
+ 15 (NomFichier) + 51 (NomComplet) + 1}
LongMes   £4+2+2+16+52+1
    LongResult
    Pushlong £LongMes        Nombre d'octets à réserver
    Pei <MyID                Numéro ramené par _MMStartup
    Pushword #%1100000000001000  Attributs mémoire
    PushLong #0
    _NewHandle
    Pulllong HandleMesCenter
    PushWord #GetMessage     (cf librairie E16.Locator)
    PushWord #fileInfoType
    Pulllong HandleMesCenter
    _MessageCenter
    Bcs DispHandMesC         Y'a pas de message.
{Y'a un message}
    Lda 3,s
    Sta <Handle+2
    Lda 1,s
    Sta <Handle
    Ldy #2
    Lda [Handle],Y
    Tax
    Lda [Handle]
    Stx <Handle+2            Adresse du message
    Sta <Handle
    Ldy #omessageData        Quel est le type de message?
    Lda [Handle],Y
    Cmp £2                   Les messages à traiter sont = 0 ou 1
    Bcs DeleteMessage        Pas connu... pas traité....
    Sta TypeMessage
{Le Message a le format suivant:
MessNext     Long   Utilisé par MessageCenter
MessType     Word   = 1 : Signifie une info concernant
                    un fichier
MessData     Block  --->
                Word   = 0 : ouverture de ce fichier
                       = 1 : Impression de ce fichier
                Str 'NomFichier'
                Str 'NomFichier Complet'
                Byte 0}
{Ajuste le nom du fichier en GSString (Longueur du nom sur 2
octets). Exemple:
0 1 2 3 4 5 6 7 8 9 A B C D
x x x x 1 0 6 F i c h i e r
Valeur = (4+2) + 1; octet longueur = 7}
    Ldy £7
    Lda [Handle],Y
    Xba
    Sta [Handle],Y
{Calcule l'adresse du nom du fichier}
    Ldx <Handle+2 Pointeur du nom
    Lda <Handle
    Clc
    Adc #7 +7 octets pour début du nom
{Les registres X et A contiennent un pointeur sur le nom du fi-
chier}
    Jsr LoadFichier
{Maintenant on teste s'il faut imprimer ce texte}
    Lda TypeMessage
    Beq DeleteMessage
{On va imprimer le fichier}
    Jsr Imprimer
{On a lu notre courrier, on peut maintenant le jeter}
DeleteMessage Anop
    PushWord #DeleteMessage
    PushWord #fileInfoType
    PushLong #0
    _MessageCenter
{On libère la mémoire utilisée par le message}
DispHandMesC Anop
    Pushlong HandleMesCenter
    _DisposeHandle
    Suite du programme
HandleMesCenter  ds 4
TypeMessage      ds 2
LoadFichier Anop      (ce passage est une esquisse)
    _OpenGS ParamOpen
    _NewHandle TailleFichier
    _ReadGS ParamRead
    _CloseGS ParamClose
    Jsr NewWindow     (ouverture d'une fenêtre
                      et passage en TextEdit)
    Rts
Imprimer Anop         (ce passage est une esquisse)
    _PrStlDialog      Option de mise en page
    _PrJobDialog      Options : qualité / Nombre de pages etc...
    _PrOpenDoc        Préparation impression
    _PrOpenPage       Préparation de la page
    _TEPaintText      Dessine le texte dans la page à imprimer
    _PrClosePage
{si plusieurs pages goto _PrOpenPage}
    _PrCloseDoc       Préparation terminée
    _PrPicFile        Imprime les pages préparées
    Rts
```

Exemple 2 : envoi d'un message MessageCenter à une autre application

```
    LongResult
    Pushlong £LongMes        Nombre d'octets à réserver
    Pei <MyID                Numéro ramené par _MMStartup
    Pushword #%1100000000001000  Attributs mémoire
    PushLong #0
    _NewHandle
    Pulllong HandleMesCenter
    Pushlong £PointMesCent
    Pushlong HandleMesCenter
    PushLong £LongMes
    _PtrToHand
    PushWord #AddMessage     (cf librairie E16.Locator)
    PushWord #fileInfoType
    Pulllong HandleMesCenter (Zone mémoire pour
                             notre message)
    _MessageCenter
```

```
PointMesCent  Anop
MessNext      ds 4     Utilisé par MessageCenter
MessType      dc.w 1   = 1 : Signifie une info concernant
                       un fichier
MessData      dc.w 1   Impression du fichier
              Str 'MonFichier'
              Str '*/MonFichier'
              Byte 0
```

Exemple 3 : envoi d'un message de type Pattern (motif)

Init Joli.Motif v1.0

```
          65816 On

MainProgram     START

                    [Variables Page Directe]

Handle     Gequ $00  ds 4    Pointeur
HandleB    Gequ $04  ds 4    Handle

MyID       Gequ $00  ds 2

                    [Programme]

[Voir commentaire dans l'exemple 4.]

   Php>>>        Sauvegarde le registre P
   Phb>>>>       Sauvegarde le registre B

   Phk           Le Registre B = Registre K
   Plb

   Long m,i      Registre en mode 16 Bits

   Tax MyId

   Lda #0
   LongResult    Pattern = 32 octets
   PushWord      Paramètres Message Center = 8
   PushWord #8+32
   Phx MyId
   Pushword #%1000000000001000
   PushLong
   _NewHandle
   Pullax HandlePattern

[Copie notre message vers le Handle pour le Window Manager]

   PushLong #MyMessage
   Pushxa
   PushLong #32+8
   _HandToPtr

                    [MessageCenter]
[Voir commentaire dans l'exemple 4]

odMessageType   Equ 2
odTypePattern   Equ 0

   PushWord #AddMessage
   PushWord #odMessageType
   PushLong HandlePattern
   _MessageCenter

   PushLong HandlePattern
   _DisposeHandle

EndNice Anop
   Plb           Restaure le registre B
   Plp           Restaure le registre P
   Rtl

HandlePattern  ds 4

                    [Couleur Mode 640]

Noir        Equ $0
BleuMarine  Equ $1
VertBout    Equ $2
VertGris    Equ $3
Rouge       Equ $4
Mauve       Equ $5
Orange      Equ $6
Rose        Equ $7
Vert        Equ $8
VertFonce   Equ $9
VertClair   Equ $A
VertTClair  Equ $B
Gris        Equ $C
Bleu        Equ $D
Jaune       Equ $E
Blanc       Equ $F

[Installe le message avec les paramètres nécessaires:
$00 : odmReserved      : Utilisé par Message Center
$04 : odmMessagetype : $0002
$06 : odmDrawType      : $0000 : Pattern
$08 : odmDraw Data     : $20 octets pour le pattern]

MyMessage   Anop
MesNext     ds 4
MesType     dc.w odMessageType
DrawType    dc.w odTypePattern

MyPattern   Anop
            dc.B Rouge*$10+Bleu,Rouge*$10+Bleu
            dc.B Jaune*$10+Gris,Jaune*$10+Gris
            dc.B Rouge*$10+Bleu,Rouge*$10+Bleu
            dc.B Jaune*$10+Gris,Jaune*$10+Gris
            dc.B Rouge*$10+Bleu,Rouge*$10+Bleu
            dc.B Jaune*$10+Gris,Jaune*$10+Gris
            dc.B Rouge*$10+Bleu,Rouge*$10+Bleu
            dc.B Jaune*$10+Gris,Jaune*$10+Gris

MaskPattern Anop
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111
            dc.B %11111111,%11111111

[Un pattern se compose d'un petit carré de 8*8 pixels. En
mode 640 chaque pixel est codé sur 2 bits, offrant donc 4 pos-
sibilités : (00 / 01 / 10 / 11). Le mélange de 2 points côte à côte
permettra d'obtenir d'autres couleurs (16) par "dithering".]

          END
```

DGA-1

La société qui diffuse la carte de digitalisation vidéo DGA-1 (voir article de Jacques Liautard dans ToolBox Mag 3) nous prie de préciser que cette carte n'est pas compatible avec les GS Roms 03, et qu'il n'y a pas de suivi matériel et logiciel de cette carte.

Dommage...

ToolBox Mag.

Exemple 4 : envoi d'un message de type Picture (dessin)

Init Joli.Ecran v1.0 en assembleur Orca

```
            65816 On
MainProgram     START
            Using Datas
                    {Variables Page directe}
Handle      Gequ $00   ds 4   Pointeur
HandleB     Gequ $04   ds 4   Handle
MyID        Gequ $00   ds 2
                    {Programme}
{A l'arrivée dans un Init, le Registre A contient un numéro
d'identificateur dispensant d'utiliser 'MemoryStartup' ou 'Get-
NewId'. Le fichier sera un TIF (temporaire Init - Type $B7). En
effet une fois le code exécuté, l'init peut être effacé de la mé-
moire, la commande Message Center que le Window Manager
appellera à chaque Startup référençant uniquement un Handle
sur la zone de l'image en mémoire. (Merci à F.Uhrich).}
    Php>>>          Sauvegarde le registre P
    Phb>>>>         Sauvegarde le registre B
    Phk             Le Registre B = Registre K
    Plb
    Long m,i        Registre en mode 16 Bits
    PushLong <HandleB   Sauvegarde les variables
    PushLong <Handle    de la Page Directe
    Sta <MyId
                    {Initialisation}
    _OpenGS OPParms     Ouvre l'image 'Ecran.Bureau'
    Jcs EndNice         Pas d'image -> quit
    Ldx Op_Filetype     Est-ce une image écran ? (type $C1)
    Cpx #$C1
    Bne CloseFile       Non -> ne rien faire
    Ldx Op_RefNum       Paramètres pour les commandes
    Stx Re_NumRef       ReadGS
    Stx Cl_NumRef       CloseGS
{Une image écran contient $8000 octets. Les paramètres pour
Message Center demandent $8 octets. On réserve donc $8008
octets.}
    Lda #0
    LongResult          Image = $8000
    PushWord            Paramètres Message Center = 8
    PushWord #$8000+8
    Pushword <MyID
    Pushword #%1000000000001000
    PushLong
    _NewHandle
    Pullax
    Bcs CloseFile       Problème de mémoire
    Stx <HandleB+2      Handle de l'image
    Sta <HandleB
    Ldy #2              Handle -> Pointeur
    Lda [HandleB],Y
    Sta <Handle+2
    Sta Re_Buffer+2
    Lda [HandleB]
    Sta <Handle
    Clc
    Adc #8              8 octets après paramètres de
                        MessageCenter
    Sta Re_Buffer       Adresse de l'image pour ReadGS
    _ReadGS ReParms     Chargement de l'image
    Bcs CloseFile       Problème à la lecture

{Installe le message avec les paramètres nécessaires:
$00 : odmReserved       : Utilisé par Message Center
$04 : odmMessage type   : $0002
$06 : odmDrawType       : $0001 : Picture information
$08 : odmDrawData       : $8000 octets de l'image}
odMessageType   Equ 2
DrawTypePict    Equ 1
    Ldy #odmMessageType
    Lda #odMessageType  Paramètre pour MessageCenter
    Sta [Handle],Y
    Iny
    Iny                 +2
    Lda #DrawTypePict   Paramètre pour MessageCenter
    Sta [Handle],Y
                    {MessageCenter}
{Message Center sera lu à chaque Startup du window mana-
ger, qui au lieu de constituer un pattern bleu comme habituel-
lement, fixera votre image sur l'écran. Voici la commande qui
fait tout le travail...}
    PushWord #AddMessage
    PushWord #odMessageType
    PushLong <HandleB
    _MessageCenter
    PushLong <HandleB   Libère la mémoire utilisée
    _DisposeHandle
CloseFile Anop
    _CloseGS ClParms    Fermeture du fichier
EndNice Anop
    PullLong <Handle    Restaure les variables
    PullLong <HandleB   de la page directe
    Plb <<<<            Restaure le registre B
    Plp <<<             Restaure le registre P
    Rtl
    END
Datas       DATA
                    {Paramètres OpenFile}
OPParms         dc.W 6
Op_RefNum       ds 2
Op_Pathname     dc.L ImageToLoad
Op_Req_Access   dc.W 0
Op_ResourceNum  dc.W 0      ; 0 : data / 1 : Fork
Op_access       ds 2
Op_Filetype     ds 2
ImageToLoad     GsString '*/System/System.Set-
                         Up/Ecran.Bureau'
                    {Paramètres ReadFile}
ReParms         dc.W 4
Re_NumRef       ds 2
Re_Buffer       ds 4
Re_Eof          dc.L $8000+8
Re_OctTrans     ds 4
                    {Paramètres CloseFile}
ClParms         dc.W 1
Cl_NumRef       ds 2
            END
```

Init JOLI.ECRAN en Orca/C

Francois Uhrich

(voir commentaires et version développée dans le source sur la disquette)

```
#pragma   optimize  -1
{les includes nécessaires}
#include   <types.h>
#include   <memory.h>
#include   <locator.h>
#include   <gsos.h>,
#include   <orca.h>

{La structure du message nécessaire à MessageCenter}
typedef struct DeskMessage {
  long    reserved;
  word    messageType;
  word    drawType;
  char    drawData[32000];
} DeskMessage,*DeskMessagePtr,**DeskMessageHndl;

void main()
{
  DeskMessageHndl mess;
  OpenRecGS openparam;
  IORecGS   readparam = { 4, 0, NIL, 32000L, 0L, 0 };
  RefNumRecGS closeparam = { 1 , 0 };
  GSString255 ecran = { 34,"*/SYSTEM/SYSTEM.SET-
  UP/ECRAN.BUREAU" };
  {Ouverture de l'image ECRAN.BUREAU}
  openparam.pCount = 6;
  openparam.pathname = &ecran;
  openparam.requestAccess  = readEnable;
  openparam.resourceNumber = 0;
  OpenGS(&openparam);
  if (toolerror()) return;
  readparam.refNum = closeparam.refNum = openpa-
  ram.refNum;

  {L'image est-elle bien de type $C1 ?}
  if (openparam.fileType==0x00C1) {
     {Allocation d'un handle de 32008
     octets pour le message}
     mess = (DeskMessageHndl) NewHandle((long)size-
      of(DeskMessage),userid(),0x8018,0L);

     if (!toolerror()) {
     {Lecture des 32000 octets de l'image dans le handle}
        readparam.dataBuffer = (*mess)->drawData;
        ReadGS(&readparam);
        {Envoi du message au tool locator }
        (*mess)->messageType = 2;
        (*mess)->drawType    = 1;
        MessageCenter(addMessage,2,mess);
     }
  }
  {Fermeture du fichier image}
  CloseGS(&closeparam);
}
```

Init JOLI.ECRAN en Orca/Pascal

Bernard Fournier

(voir commentaires dans le source sur la disquette)

```
{$keep 'JoliEcran'}
{$optimize -1}

PROGRAM JoliEcran;
uses Common,ToolLocator,MemoryMgr,
MscToolSet,GSOS;
Label 100;

Const noError =0;
    chaine  =2;
Type
  MyMessageHndl     = ^MyMessagePtr;
  MyMessagePtr      = ^MyMessage;
  MyMessage     = record
                MReserved: LongInt;
                MType  : Integer;
                MDraw  : Integer;
                MData  : packed array [1..32000] of byte;
            end;

VAR
 MyMemoryID     : integer;
 CustomScreen   : MyMessageHndl;
 OpenName       : gsosInString;
 FileOpen       : openOSDCB;
 FileRead       : readWriteOSDCB;
 FileClose      : closeOSDCB;

begin
 MyMemoryID:= MMStartUp;
 if ToolError = noError then
  begin
  CustomScreen:=MyMessageHndl(NewHandle(32008,
  MyMemoryID,attrLocked,nil));
   with CustomScreen^^ do begin
   MReserved   :=0;
   MType       :=2;
   MDraw       :=1;
   end;

  with OpenName do begin
      theString:='*/System/System.SetUp/Ecran.Bureau';
      size:=length(OpenName.theString);
      end;
  FileOpen.pcount:=6;
  FileOpen.pathname:=pointer(@OpenName);
  FileOpen.requestAccess:=0;
  FileOpen.resourceNumber:=0;
  OpenGS(FileOpen);
  if ToolError<>NoError then goto 100;
  if FileOpen.fileType<>$C1 then goto 100;
  FileRead.pcount:=4;
  FileRead.refnum:=FileOpen.refnum;
  FileRead.databuffer:=@CustomScreen^^.MData;
  FileRead.requestcount:=32000;
  ReadGS(FileRead);
  if ToolError<>NoError then goto 100;
  MessageCenter(1,2,Handle(CustomScreen));
 end;

100:
 FileClose.pcount:=1;
 FileClose.refnum:=FileOpen.refnum;
 CloseGS(FileClose);
 DisposeHandle(handle(CustomScreen));
end.
```

C facile 4 : écrire des Inits en C

François Uhrich

Les puristes vont hurler mais aujourd'hui il y a un peu d'Assembleur au menu! C'est qu'il s'agit de demander au compilateur C de faire un type de programme qu'il n'a pas prévu: un Init.

Qu'est-ce qu'un Init ?

Les Inits de l'Apple IIgs sont des programmes situés dans le répertoire */SYSTEM/SYSTEM.SETUP. Ces programmes sont exécutés pendant le boot après le chargement de GS/OS et des drivers, mais avant les accessoires de bureau (NDA et CDA). Il y a deux types d'Inits:

- les **Inits permanentes** (fichier de type $B6 ou PIF) qui sont chargées, exécutées et qui restent en mémoire. Cela peut être le cas par exemple d'Inits patchant les tools, de l'affichage d'une horloge dans la barre de menu.
- les **Inits temporaires** (fichier de type $B7 ou TIF) qui sont chargées, exécutées puis purgées de la mémoire. Ces Inits ont une opération à effectuer et après cela leur présence est inutile: l'affichage d'une image ou le déclenchement d'un son au boot sont des exemples d'Inits temporaires.

Quelle(s) différence(s) y a-t-il entre un programme "classique" (de type S16) et une Init (temporaire ou permanente)? Réponse: *"presque"* aucune !

Comme tout autre programme sur GS, une Init peut utiliser les tools du système, utiliser GS/OS... etc. Un détail à souligner: les accessoires de bureau sont absents au moment de l'exécution d'une Init.

La seule différence, mais elle est capitale, c'est qu'un programme S16 se termine par un appel au Quit de GS/OS, alors qu'une Init doit se terminer par un simple retour long de sous-programme: RTL.

Nous allons voir comment réaliser des Inits avec Orca/C et APWC.

Les Objets de la Compilation Orca/C

Une fois un programme source Orca/C compilé, deux fichiers de type OBJ sont créés:

xxx.root et *xxx.a* (*xxx* étant le nom de votre programme).

xxx.a contient la compilation de votre programme source, alors que *xxx.root* est un fichier standard créé automatiquement par le compilateur.

Il y a trois versions différentes de *xxx.root* suivant qu'il s'agit d'un programme classique S16 ou EXE, d'un NDA (commande pré-processeur *#pragma NDA*) ou enfin d'un CDA (commande pré-processeur *#pragma CDA*). Dans le cas d'un accessoire de bureau, *xxx.root* contient en fait l'en-tête nécessaire à un NDA ou à un CDA. Comme il nous manque un *#pragma INIT*, nous devrons nous rabattre sur le *xxx.root* classique:

```
            case on
~_ROOT      start
            pea $2000
            jsl ~_BWSTARTUP
            pea ~GLOBALS|-8
            plb
            plb
            jsl ~C_STARTUP
            jsl main
            jsl ~C_SHUTDOWN
            end
```

~_BWSTARTUP est une fonction de la librairie *SYSLIB* qui initialise l'environnement C ainsi que l'espace nécessaire à la pile (pea $2000 pour réserver 8Ko de pile) et enfin l'ouverture de SANE (outil de calcul arithmétique).

~C_STARTUP (librairie *ORCALIB*) s'occupe essentiellement de la lecture de la ligne de commande éventuelle à passer en argument *argc* et *argv* au *main()* (voir C Facile Toolbox Mag n°2).

La fonction principale *main()* de votre programme est enfin exécutée.

~C_SHUTDOWN (librairie *ORCALIB*) termine l'exécution en purgeant la mémoire allouée pour la pile et par le programme, et enfin appelle la fonction *~QUIT* qui exécutera le QUIT de GS/OS:

```
GSOS        gequ $E100A8
~QUIT       entry
            pha
            jsl ~MM_INIT
            PushWord >~USER_ID
            _DisposeAll
            pla
            jsl >GSOS
```

```
            dc  i2'$0029'        Quit
            dc  i4'~QUITPARAM'
~QUITPARAM anop
            dc  i4'0'
            dc  i2'0'
            end
```

Comme l'a très bien décrit Mike Westerfield (de Byte Works Inc., en fait *l'auteur* d'Orca Pascal et Orca C) dans son article concernant les inits en Orca/C ou Orca/Pascal dans le numéro d'avril 1990 de l'excellent magazine 8/16, il ne suffit pas de supprimer le QUIT GS/OS de *~QUIT*.
En effet il y a problème avec la pile. Lorsque GS/OS reçoit un QUIT, peu importe l'état de la pile, le programme courant est abandonné pour passer le contrôle à un autre. En revanche avec un RTL, il faut que la pile soit impeccable! Or, dans un programme en C, l'instruction *exit()* permet de quitter un programme à partir d'un sous-programme quelconque et Orca/C dans ce cas ne maintient pas sa pile en bon état. La solution de Mike Westerfield est donc de stocker la position de la pile à l'arrivée dans *~_ROOT* et de restaurer cette position dans *~QUIT* avant un simple RTL.
Voici le ~_ROOT pour les Inits:

```
            case on
~_ROOT      start
            tax
            tsc
            sta  >~QUITSTACK
            txa
            pea  $2000
            jsl  ~_BWSTARTUP
            pea  ~GLOBALS/-8
            plb
            plb
            jsl  ~C_STARTUP
            jsl  main
            jsl  ~C_SHUTDOWN
            end
```

C'est *~QUITSTACK* qui contiendra la position de la pile.
A noter une *importante modification* par rapport à la version de Mike Westerfield: à l'arrivée dans *~_BWSTARTUP*, le registre A ne doit pas avoir été modifié car le loader a la bonne idée d'y avoir stocké l'identificateur mémoire du programme et cela également dans le cas des Inits (voir les File Type Notes $B6 et $B7 d'Apple). *~_BWSTARTUP* commence donc par stocker A (après toutefois un ora $0100) dans la variable *~USER_ID* qui sera notamment utilisée par la

Les Structures de Données en C

Dans le source de Joli.Ecran, la structure du message contenant l'image du bureau est définie ainsi:

```
typedef struct DeskMessage {
  long    reserved;
  word    messageType;
  word    drawType;
  char    drawData[32000];
} DeskMessage,*DeskMessagePtr,**DeskMessageHndl;
```

Après cette déclaration de type, le programmeur peut librement utiliser ces nouveaux types de données comme n'importe quels autres types:

```
DeskMessage message;            /* structure */
DeskMessagePtr messagePtr;      /* pointeur sur structure */
DeskMessageHndl messageHndl;    /* handle (pointeur de pointeur) sur structure */
```

Pour accéder à partir de ces variables à chacun des différents champs de la structure, il faut utiliser le point '.' ou bien '->' dans le cas d'un pointeur sur la structure:

```
message.messageType = 2;
messagePtr->drawType = 1;
(*messageHndl)->drawData[0]=0;
```

La macro instruction *sizeof* du pré-processeur C permet de connaître la taille en octets d'un type ou d'une variable quelconque. Ainsi:
sizeof(int) vaut 2.
sizeof(long) vaut 4.
sizeof(char) vaut 1.
sizeof(DeskMessage) vaut 4+2+2+32000=32008 octets.
sizeof(message) vaut également 32008.
mais *sizeof(messagePtr)* et *sizeof(messageHndl)* ne valent que 4, puisque ce sont des adresses...

fonction *userid()* de la librairie ainsi que pour un *DisposeAll* final. Le registre A a donc tout intérêt à être valide... Voilà la raison du *tax* et du *txa* dans *~_ROOT*.
La fonction *~QUIT* (segment *~_BWCommon* de la librairie *SYSLIB*) devient alors:

```
~QUIT       entry
            pha
            jsl  ~MM_INIT
            PushWord >~USER_ID
            _DisposeAll
            plx
            lda  >~QUITSTACK
            tcs
            txa
            rtl
~QUITSTACK entry
            ds  2
```

Sur la disquette vous trouverez donc *CInit.root* et *Common.root*. Voici les opérations à effectuer afin d'obtenir votre Init que je nomme *MonInit.cc:*
• compilation:
compile MonInit.cc keep=$
• remplacement de *MonInit.root* par la version modifiée *CInit.root*:
copy -c CInit.root MonInit.root
• édition de liens en plaçant *Common* (contenant le nouveau *~QUIT*) avant la librairie :
link MonInit Common 2/orcalib 2/syslib keep=MonInit

Les Objets de la Compilation APWC

Avec APWC il y a quelques différences.
Tout d'abord, il n'y a qu'un seul fichier OBJ *xxx.root* issu de la compilation. Il correspond donc au *xxx.a* de Orca/C. C'est au moment de l'édition de liens qu'un en-tête est ajouté:
link 2/start MonProg 2/clib keep=MonProg
Le *start.root* situé dans le répertoire des librairies correspond au *xxx.root* de Orca/C et effectue à peu près les mêmes opérations d'initialisation et de fermeture. Le quit GS/OS est réalisé par la fonction *CQuit* de la librairie *CLIB* appelée par *start.root*:

```
GSOS        gequ  $E100A8
CQuit       start
            lda  $04,s
            sta  $0014
            lda  $06,s
            sta  $0014+$0002
            lda  $08,s
            jsl  >GSOS
            dc   i2'$0029'        Quit
            dc   i4'$00000000'
            rtl
            end
```

Avec APWC, la position de la pile ne pose aucun problème (visiblement *start.root* a été écrit en C comme le démontre le RTL après le QUIT GS/OS...), par conséquent la solution la plus simple sans modifier *start.root* (ce qui pourtant permettrait de gagner pas mal d'octets en APWC... avis aux amateurs !) est de **linker une fonction *CQuit* "vide"** avant la librairie *CLIB*:

```
CQuit start
       rtl
       end
```

Sur la disquette, vous trouverez donc *CQuit.root*. Voici les opérations à effectuer afin d'obtenir votre Init que je nomme *MonInit.cc* écrite cette fois-ci avec APWC:
• compilation:
compile MonInit.cc keep=$
• édition de liens en plaçant CQuit avant la librairie CLIB:
link 2/start MonInit CQuit 2/CLib keep=MonInit

Un exemple: Joli.Ecran en C

Vous trouverez sur la disquette le source C (comme d'habitude à la fois pour Orca/C et APWC) de l'Init Joli.Ecran (Nice.Screen) que Patrick Desnoues vous présente en Assembleur dans ce numéro.
Attention, le source de Joli.Ecran en Orca/C imprimé page 9 de ce numéro est une version abrégée pour permettre la comparaison avec la version en Assembleur et en Pascal. Pour avoir le source complet, avec les commentaires et une version pour APW/C, imprimez le fichier-source de la disquette.

La version APWC occupe après édition de liens 11 blocs, alors que Orca/C n'occupe que 5 blocs (ExpressLoad inclus dans les deux cas). Si je n'avais pas utilisé des appels directs à GS/OS pour l'ouverture (OpenGS), la lecture (ReadGS) et la fermeture (CloseGS) du fichier d'image, mais plutôt des appels à la librairie standard C (*stdio.h* avec *fopen*, *fread* et *fclose*) l'occupation sur disque aurait au moins doublé... La puissance de la librairie standard C d'entrée-sortie ne devient intéressante que dans le cas de traitement complexe de fichiers mais certainement pas pour la simple ouverture, lecture, fermeture d'un unique fichier.

A noter: pour gagner de la place sur le disque, on pourrait modifier Joli.Ecran pour qu'il charge aussi une image compactée de type $C0 (un fichier-écran $C1 fait 32Ko *[NDLR: mais pas forcément 65 blocs, voyez la disquette...]*).

8/16 Central: P.O. Box 11250
Overland Park, KS 66207, USA
Abonnement un an (12 disquettes): $69.

Ecrire des Inits en Pascal

Bernard Fournier

L'Init "Joli.Ecran" de Patrick Desnoues se résume à un appel à GS/OS et un appel à la ToolBox (MessageCenter). Le traduire en Pascal, apparemment, rien de plus facile. Et pourtant...

Un Init en Pascal, est-ce possible ?

En Pascal, le compilateur et l'éditeur de liens ajoutent en début de programme un code servant à initialiser certains paramètres (positionner le registre de pile, de banc, réserver la mémoire pour les variables...) et à la fin un code permettant de revenir au programme ayant appelé l'application (libérer la mémoire réservée et effectuer un QUIT_GSOS).
Ces lignes de code sont ajoutées automatiquement à toute application, et nous sommes obligés de passer par là, puisque, s'il existe des directives spéciales de compilation pour les NDA et les CDA, il n'en existe pas pour les Inits.
Mais un Init n'est pas une application: lorsqu'il est exécuté, il doit rendre la main au Loader qui continuera le chargement du système d'exploitation (par un RTL). Or, le code standard généré par le compilateur effectue un QUIT_GSOS.

Il faut donc intervenir dans le processus de compilation pour le modifier. Conséquence: **tout ce qui va suivre ne peut concerner que ORCA/Pascal et non TML. Comme TML ne permet pas d'intervenir dans le processus de compilation/linkage, il ne permet pas de programmer des Inits.**

Le compilateur Orca/Pascal génère deux fichiers objets: le fichier *.A* contenant le code compilé proprement dit, et le fichier *.ROOT* contenant les initialisations de début et fin ainsi que le QUIT_GSOS. L'éditeur de liens va ensuite lier ces deux modules pour constituer l'application. Pour un Init, l'idée principale consiste à compiler notre propre fichier *.ROOT*, ensuite on lie avec *COMMON.ROOT* et le tour est joué!

Comment procéder ?

La méthode à suivre, avec les explications complètes, a été publiée dans la revue Américaine 8/16 du mois de Mai 1990 par Mike Westerfield.
Note: sur la disquette ToolBox Mag 4 se trouvent les sources et macros de *COMMON.ASM* et *PASINIT.ASM* ainsi que leurs codes objets *COMMON.ROOT* et *PASINIT.ROOT*. *Les deux sources .ASM sont particuliers à ORCA/Pascal;* ceux relatifs à ORCA/C diffèrent notablement. J'ai inclus la correction de François Uhrich au bug du source de Mike Westerfield (voir C Facile 4 dans ce numéro).

Méthode à suivre

- générer les macros à partir du fichier *COMMON.ASM -> COMMON.MAC*.
- compiler *COMMON.ASM -> COMMON.ROOT*.
- compiler le noyau *PASINIT.ASM -> PASINIT.ROOT*.
- compiler le source de l'init *INIT.PAS -> INIT.ROOT* et *INIT.A*.
- remplacer le noyau standard *INIT.ROOT* par notre noyau personnalisé *PASINIT.ROOT*.
- linker le total: l'éditeur de liens va alors utiliser les routines de *INIT.A* ainsi que *COMMON.ROOT* et *INIT.ROOT*(qui en fait est la copie de *PASINIT.ROOT*) pour effectuer l'initialisation et le quit (un RTL) de notre Init.

Avec cette méthode, vous pourrez compiler n'importe quel Init de votre cru en Orca/Pascal, pourvu que vous utilisiez bien *COMMON* et *PASINIT* tels qu'ils sont fournis sur la disquette ToolBox Mag 4, et que vous respectiez le processus de compilation/substitution/linkage tel qu'indiqué dans le script *MAKINIT*.

Remarque:
Dans ToolBox Mag 2, François Uhrich parlait d'un utilitaire *MAKE* permettant d'effectuer des processus conditionnels liés aux dates de mise à jour des fichiers sources. J'ai repris cette méthode dans le script *MAKEFILE*. Ceux qui possèdent cet utilitaire utiliseront donc *MAKEFILE*, alors que ceux qui n'ont que les utilitaires ORCA standard devront lancer le script *MAKINIT*.
Bien évidemment, chacun devra adapter les préfixes utilisés pour les accès aux librairies assembleur.
Ceux qui n'ont pas l'assembleur ORCA devront utiliser le script *MAKPAS* qui emploie les modules objets et non les sources des routines assembleur.

Exotica

Des nouvelles du monde extérieur

par François Hermellin

Multimédia : l'explosion

Le multimédia est en pleine ébullition ces temps-ci. **MacroMind**, qui diffuse le meilleur produit multi-média (Director), a annoncé sa volonté de produire une version qui permettrait aux possesseurs d'OS/2 de rejouer des animations créées avec Director. Jusqu'à présent, les présentations réalisées par ce programme ne pouvaient tourner que sur un PC sous Windows 3.0 ou sur un Mac. La société compterait proposer des versions pour Amiga et pour des plates-formes UNIX (Bonjour, Steve !).

A côté des applications sur micro-ordinateurs, MacroMind Director, HyperCard 2.0 ou... ToolBox (non, l'autre, le logiciel qui tourne sur PC), les solutions spécifiques montrent le bout de leur nez.

Le **CD-Interactif**, développé par **Philips et Sony**, devrait être lancé dans le grand public en septembre prochain. Après avoir connu une période de retrait volontaire de 1987 à 89 (avant cette date, le CD-I ne pouvait animer une image que sur 1/4 de l'écran, alors que son concurrent le DVI était en 'Full motion'), Philips repart à l'attaque. **Motorola** a produit pour le CD-I une puce intégrée VLSI permettant des animations plein écran. Matsushita, un des plus gros géants de l'électronique japonais, s'est allié au consortium Philips/Sony en Mai 1989.

Techniquement, le CD-I gère quatre niveaux de compressions sonores autorisant des qualités et des durées plus ou moins importantes (d'une heure en qualité CD, à plus de neuf heures et demie, avec un taux de compression élevé, mais une qualité juste bonne pour de la voix). En matière de vidéo, la couleur est codée sur 5 bits par pixel, avec une palette de couleurs de 32 000 teintes. Chaque CD-I contient 650 Mégas de données.

Le lecteur grand public (CDI 602) contient un lecteur de CD-I, compatible CD audio, et un lecteur de disquettes 3,5'. Il est pourvu d'1 Mo de RAM et de 8 Ko de RAM non-volatile. Il gère les standards TV NTSC et PAL, et contient deux slots d'extensions. On peut adjoindre à ce système une unité de contrôle multimédia (pour gérer d'autres sources tels qu'un un lecteur CD-Vidéo...).

Des kits de développement sont disponibles pour Mac, PC et stations de travail Sun. Les premiers logiciels disponibles sont soit des encyclopédies (Encyclopédie Grolier, Banque de photographies de Time-Life, Le monde à travers les timbres...), soit des applications pour tout petits (Une visite rue Sésame, le Livre des coloriages...), soit enfin des jeux (Dark Castle en 3-D, Wings, Déjà vu...).

De l'autre côté, l'éternel rival de Motorola, **Intel**, qui a investi 400 Millions de dollars (2 400 Millions de francs environ) en R & D en 1990, prépare depuis plusieurs années le **DVI (Digital Video Interactive)**. La solution de cette société est un peu différente. Constructeur de circuits intégrés, Intel se contente de proposer aux constructeurs de micro-ordinateurs des puces à utiliser au sein de leurs machines.

Le i750 comprend deux composants : le processeur de pixels 82750 PB et le processeur d'affichage 82750 DB. Le processeur de pixels s'interface à la Ram vidéo pour compresser, stocker et rechercher les données. Il permet des animations de qualité en compressant et décompressant chaque trame d'informations en moins de temps qu'il ne faut pour l'afficher. Le processeur d'affichage affiche les images dans les formats VGA, NTSC, PAL/SECAM, et génère les signaux de synchronisation et de commandes d'écrans. Le coût de ces circuits est de l'ordre de 400 F pièce. Une machine intègre déjà ces circuits : il s'agit du **Cornucopia**.

Enfin, **Commodore**, avec des moyens largement plus limités, propose sa solution. Baptisé **CDTV**, il est fondé sur un concept plus large que celui de ses concurrents. Le but du CDTV serait de remplacer d'autres sources de données numériques telles que les lecteurs de CD Audio, CD Vidéo... Les quelques applications présentées ne font pas assez la différence entre le CDTV et l'Amiga pour qu'actuellement l'appareil apparaisse réellement intéressant.

Réseaux

Après plusieurs années de bons et loyaux services de ce bon vieil AppleTalk, Apple a finalement décidé de passer à la vitesse supérieure (10 Mo par seconde) pour les réseaux: LocalTalk va peu à peu être remplacé par **Ethernet**. Apple propose une série de cartes pour le Mac LC (2 090 F HT) et les Mac II (carte Nubus possédant son propre processeur 68 000). Ces dernières seront utilisables sur tous les types de média (câble fin et paires torsadées).

Claris : Pro ?

Claris, après des rapports un peu compliqués avec Apple (contrôlée par le constructeur à la pomme, puis indépendante, et de nouveau sous le giron d'Apple!), s'apprête à lancer sur le marché des versions améliorées, dites "Pro", de ses logiciels vedettes. **FileMaker Pro**, l'excellent gestionnaire de fichiers, est disponible depuis quelques mois.

Mac Draw II Pro a été annoncé à la Mac World Expo. Il propose de sérieuses améliorations (plus d'une centaine) : gestion de la couleur et des dégradés, intégration d'un véritable traitement de texte, courbes de Bézier et outil de présentation de diapositives.

Une **version Pro de Mac Write II** vient d'être annoncée. Elle devrait incorporer la création de tableaux et la gestion d'index. Cependant, il est en cours de développement et toutes ses caractéristiques n'ont pas été encore fixées définitivement.

Nouveaux Macs

Les nouveaux Mac se vendent bien, très bien, trop bien même car Apple n'arrive pas toujours à suivre la demande. A la surprise du constructeur, beaucoup d'achats ont été le fait de sociétés (PME-PMI et grands comptes). Les utilisateurs individuels ne sont cependant pas en reste. Conséquence de ce succès commercial, des éditeurs pas, ou peu présents sur le marché du Mac, regardent désormais cette machine d'un tout autre œil. Cela est particulièrement vrai dans le marché des jeux. Les éditeurs anglais, traditionnellement absents sur cette machine, s'intéressent désormais de très près au Mac. De même, les premières productions US débarquent. Première surprise: ce sont des jeux de qualité. On peut ainsi jouer sur son LC ou son SI avec Indiana Jones ou Dragon's Lair en 256 couleurs (alors que les versions Amiga ou ST n'en possèdent que 16, eh, eh!).
Le marché des upgrades pour les nouveaux Mac est en plein boom. Plusieurs dizaines de sociétés proposent des solutions hard pour "doper" les Classic, SI, LC. C'est à qui proposera l'extension mémoire la plus rapide et la moins chère, l'écran 15', pivotant ou non, la carte accélératrice... Du nouveau aussi prochainement chez Apple dans le domaine des périphériques (imprimantes, écrans...) et des unités centrales.

Vrais Portables

Le marché des portables (vraiment portables, et non pas transportables, si vous voyez à quel portable je fais allusion...) n'en finit plus d'exploser. Tous les constructeurs travaillent sur des modèles style Carnets de Notes, en recherchant des solutions aux principaux problèmes qui se posent, à savoir l'autonomie des batteries, le poids du micro et la qualité de l'écran. La dernière trouvaille dans ce domaine concerne le stockage des données. Après les mémoires à bulles, les disques durs de 2,5', les drives de 2,88 Mo, la dernière nouveauté réside dans **les cartes à mémoire Flash**. Ces mémoires d'Intel à semiconducteurs pèsent 28 g, ont des temps d'accès de 8 Mégaoctets par seconde et existent en version de 1 Mo ou 4 Mo.

Quelques chiffres européens

• Parts de marché en volumes en France (juillet-Septembre 1990, enquête Dataquest) : Apple 11,6 %, IBM 11,2 %, Compaq 8,6 %, Commodore 7,2 %, Goupil 6,7 %, Atari 6,7 %, Bull 6,4 %, Autres 41,7 %.
• Apple France a terminé son exercice fiscal 90 avec une hausse de son Chiffre d'Affaires de 29 % soit 2,7 Milliards de francs (alors que celui d'Apple US n'a augmenté que de 5 %). En Europe, Apple a amélioré son CA de 28 % (1,57 milliard de dollars).

Sur le front des imprimantes

• Au Comdex de Las vegas, on pouvait découvrir les premières **imprimantes compatibles TrueImage**. TrueType doit être intégré dans la version 3.1 de Windows. Selon certaines rumeurs, il se pourrait qu'Apple livre bientôt, avec son **imprimante à bulle d'encre 360 dpi** dont le prix tournerait autour de **3 000 F HT**, l'init TrueInit qui permettrait de se servir de TrueType.
• La première **imprimante Postscript level 2** existe. Elle s'appelle LZT 660 et est fabriquée par Dataproducts ($ 2995). Parmi toutes les améliorations de Postscript, on peut en citer une que beaucoup de monde attendait : la possibilité de gérer des fichiers compressés/décompressés.

Rouge = Rouge ?

La recherche d'une restitution correcte de la couleur dans toute la chaîne graphique (du scanner couleur à l'imprimante couleur en passant par l'écran) est le dernier dada de la plupart des sociétés américaines travaillant dans ce domaine. En effet deux rouges annoncés comme identiques à travers leur référence du nuancier Pantone™, ne seront pas exactement semblables sur deux écrans différents, ni de l'écran au papier imprimé. Une première solution a consisté en la mise au point de **calibrateurs** par des sociétés comme **Radius**. Ces petits appareils se collent à l'écran et analysent la différence des teintes perçues par rapport aux couleurs d'une mire. **Kodak** vient de proposer toute une série d'utilitaires et d'outils de développement pour garantir que la couleur à l'écran sera identique à la hard copy. Cette nouvelle technologie porte le doux nom de **Photo YCC** et consiste en une méthode innovante pour représenter la couleur sous une forme digitale.

Clône 386

Le monopole de la production de 386 d'Intel semble prendre fin. AMD semble avoir mis deux équipes sur le coup. La première étudiait la logique du composant vedette (la plus forte progression de vente pour les années à venir) d'Intel, alors que la seconde décortiquait physiquement la puce (ce que l'on appelle chastement "Reverse engineering"). Le processeur clône d'AMD serait même plus performant que l'original (consommation d'énergie moins importante, vitesse de l'horloge pouvant passer à 0 Mhz en cas de non utilisation momentanée sur un portable par exemple...). Pour le moment, aucune décision juridique n'a été prise concernant la vente de cette puce, dans un sens comme dans l'autre. Il existe seulement un jugement qui autorise AMD à utiliser '386' dans le nom de ses composants.

La ruée vers l'Est

• Le Japon est, pour Apple, un marché qui explose. 1990 a été l'année où les Japonais ont véritablement croqué dans la pomme. Certains prévoient même que la croissance des parts de marché dans l'archipel dépassera celle de la France ou de l'Europe.
• Le gouvernement US a finalement décidé d'éliminer tout contrôle sur les exportations de matériel basé sur le 486 à destination du pays de la vodka et de la révolution prolétarienne.

Apple Service Team...

Apple USA vient d'ouvrir une **ligne téléphonique gratuite** pour "résoudre les problèmes de (nos) clients". Ce n'est pas une hot-line technique , mais "une assistance pour les personnes qui n'arrivent pas à obtenir entière satisfaction de leurs concessionnaires" d'après John Cook. Et moi qui croyais qu'il n'y avait qu'en France que les concessionnaires n'étaient pas parfaits !

Mac Eraser : le source

Olivier Goguel

*[NDLR: Lecteurs, savourez vos privilèges. ToolBox Mag, qui vous avait déjà offert un Tool du "Pape du No-Tools", qui vous offre un super-jeu du même auteur dans ce numéro même, réussit le **tour de force historique** de publier **un source vraiment commenté d'Olivier Goguel !***
Rappelons que le programme "Mac Eraser" (cf ToolBox Mag 3) permet au GS de faire ce que le Mac ne sait pas faire: effacer ses disquettes sans les reformater. Le source Merlin va vous montrer entre autres comment lire et écrire les blocs de 524 octets - les 512 habituels plus les 12 de "signatures" (tags) - des disquettes Macs... sans Mac.]

MAC ERASER v1.0
Olivier Goguel

```
        ORG $800
        XC
        XC

        DUM $00
Msg     DS 2
        DEND

        SEC
        XCE
        SEI          pas d'interruptions autorisées
        STA $C000    Initialisations
        STA $C00C
        JSR $FE93
        JSR $FE89
        JSR $FB2F
        JSR $FC58
        STA $C00F

        INC S03F4    Force un reboot si on appuie sur
                     Ctrl-Ó-Reset, pour éviter de
                     rester avec des appels trafiqués.

        JSR Print
        DFB 1
        ASC "    MAC ERASER v1.0",8D
        ASC "    "
        FLS "LLLLLLLLLLLLLL",00  MouseText
        JSR Print
        DFB 20
        ASC "     TODAY'S QUOTE :",8D
        FLS "SSSSSSSSSSSSSSSSSSSSS"
        FLS "SSSSSSSSSSSSSSSSSSS"
        ASC " FOR EVERY MACINTOSH SOLD",8D
        ASC " THERE IS AN IDIOT BORN !",00  si!
```

Déconnecte un éventuel Diversi-Cache (ou autre routine barbare) en remettant les valeurs par défaut des vecteurs du smartport

```
        LDX #0
]lp     LDAL $E10F6F+4,X
        STAL $E10F6F,X
        INX
        LDAL $E10F6F+4,X
        STAL $E10F6F,X
        INX
        LDAL $E10F6F+4,X
        STAL $E10F6F,X
        INX
        LDAL $E10F6F+4,X
        STAL $E10F6F,X
        INX
        INX
        INX
        INX
        INX
        CPX #8*8
        BNE ]lp
```

Tout ce qui concerne la lecture/écriture de blocs de 524 octets se trouve réuni ici.
*Les routines de la Rom peuvent par défaut lire/écrire des blocs de 512 ou 524 octets. Il y a en théorie un appel permettant de travailler avec la longueur voulue: mais l'absence de documentation officielle **complète** du smartport fait que cet éventuel appel ne se trouve documenté nulle part. **Heureux possesseur de la documentation du drive 3.5, pense aux autres...***
Puisque l'on ne peut pas dire proprement au smartport que l'on veut travailler avec des blocs de 524, on va lui forcer la main "sauvagement".
En examinant un tant soit peu les routines du smartport, et en particulier les routines de post/pré-nibblisation, on rencontre un drapeau ($E10F51) qui indique au smartport la taille des blocs. S'il est à $00, on va travailler en 512 octets par bloc, s'il est à $FF on travaillera avec des blocs de 524 octets.
Un de mes premiers patches a été de forcer ce drapeau à $FF avant chaque appel smartport. Cette méthode relativement simple à mettre en œuvre ne marche pas: le smartport initialise ce drapeau à $00 à chaque nouvel appel.
On va donc être obligé de forcer ce drapeau, mais à un niveau plus bas dans le smartport. Pour cela, on va détourner un des vecteurs du smartport, en l'occurrence celui situé en $E10FA7, qui correspond à un des dispatchers principaux du smartport, et envoie, selon l'appel demandé, aux routines de lecture/écriture. On va tout simplement forcer, en détournant ce vecteur, le drapeau $E10F51 à $FF, puis continuer l'exécution normale du smartport en se rebranchant sur l'ancienne adresse détournée.

```
        LDAL $E10FA8      On sauvegarde l'ancienne
        STA NormalVect+1  valeur du vecteur concerné
        LDAL $E10FA9      pour pouvoir la restaurer
        STA NormalVect+2  lorsqu'on voudra quitter.
        LDAL $E10FAA
        STA NormalVect+3

        LDA #<Detour      On met à la place
        STAL $E10FA8      l'adresse de notre
        LDA #>Detour      routine.
        STAL $E10FA9
        LDA #^Detour
        STAL $E10FAA

        BRA Continue
```

La routine suivante sera exécutée à chaque appel de lecture/écriture par le smartport.

```
        MX %11
```

La routine est appelée avec : B=$E1 et x=1, m=1

```
Detour LDA #$FF    Force le drapeau $E10F51
       STA $0F51   à $FF (B=$E1 à l'appel)
NormalVect JML $000000  et on retourne normale-
                        ment au smartport...
```

La suite du programme ne présente plus de difficulté.

```
Continue JSR EjectDisk

]mainloop JSR Print
        DFB 9
        ASC "    PRESS Q TO QUIT ",8D8D
        ASC "            OR   ",8D8D
        ASC "    INSERT ANY DISK"
        ASC " TO BE ERASED      ",00

WaitKey STA $C010

]lp     LDA $C000          scrute le clavier
        BPL NoKey

        CMP #"Q"
        BEQ GoQuit         Appuie sur Q : Quit
        CMP #"q"
        BNE WaitKey

GoQuit BRL Quit

NoKey JSR GetEntry
        BEQ ]lp      Attente de l'insertion d'un disque

        JSR Print
        DFB 9
        ASC "    ERASING DISK...",8D8D
        ASC "            ",8D8D
        ASC "                    ",00

        JSR Erase_Disk     Effacement du disque
        BCS SomePbm
        BRL NoPbm

SomePbm LDX #200

        CMP #$2B
        BNE NoWriteProtect

        JSR Print
        DFB 13
        DFB "G"&$9F
        ASC "DISK WRITE PROTECTED:"
        ASC " OPERATION CANCELED"
        DFB "G"&$9F,00

        BRA ContErr

Translate CLC
        ADC #"0"
        CMP #"9"+1
        BCC Chiffre
        ADC #"A"-"9"-2
Chiffre RTS

NoWriteProtect PHA
        AND #$F0
        LSR
        LSR
        LSR
        LSR
        JSR Translate
        STA ErrCode
        PLA
        AND #$0F
        JSR Translate
        STA ErrCode+1
        JSR Print
        DFB 13
        DFB "G"&$9F
        ASC "AN ERROR S"
ErrCode ASC "?? OCCURS: OPERATION CANCELED."
        DFB "G"&$9F,00
        BRA ContErr

NoPbm LDX #2       Temps d'attente

        JSR Print
        DFB 13
        ASC " OPERATION SUCCESSFULL...",00

ContErr JSR EjectDisk

]lp     LDA $C019  NDLR: VBL; vraiment utile ici?
        BPL ]lp
]lp     LDA $C019
        BMI ]lp
        BRL ]mainloop
```

```
Quit    JSR $FB2F
        JSR $FC58          home
```

Avant de sortir, on enlève notre détournement en remettant la valeur du vecteur par défaut. On risquerait d'avoir des ennuis si on continuait de travailler avec des blocs de 524 octets.

```
        LDX #3
]lp     LDAL $E10FAB,X
        STAL $E10FA7,X
        DEX
        BPL ]lp

        LDA #0
        STAL $E10F51

        CLI

        JSR $BF00          Prodos MLI
        DFB $65            Appel Quit
        DA QuitParm
QuitParm HEX 04
        DS 6
```

```
Eject_Parm HEX 030100000400   éjecte slot 5 drive 1

Wait_Parm HEX 0301
        DA Entry
        DS 2
Entry   DS 1
        HEX 000000
```

```
EjectDisk JSR $C50D        appel SmartPort
        HEX 04
        DA Eject_Parm
        JSR GetEntry
        BNE EjectDisk
        RTS
```

```
GetEntry JSR $C50D        attend l'introduction
         HEX 00           d'un disque (appel status)
         DA Wait_Parm
         LDA Entry
         AND #$10         un seul bit nous intéresse
         RTS
```

```
Erase_Disk STZ Block
         STZ Block+1

         LDA #<Free
         STA Mem
         LDA #>Free
         STA Mem+1
```

On travaille tout-à-fait normalement avec les blocs de 524 octets en faisant des appels totalement classiques.

```
]loop JSR $C50D           smartport
         DFB 02           Write
         DA ReadParm
         BCC NoErr
         RTS

NoErr LDA Mem
         CLC
         ADC #<524
         STA Mem
         LDA Mem+1
         ADC #>524
         STA Mem+1

         INC Block
         LDA Block
         CMP #$1D         ça suffira
         BNE ]loop
         CLC
         RTS

ReadParm HEX 03
         HEX 01
Mem      DA 0
Block ADRL 0
```

```
Print    PLA              routine d'affichage
         STA Msg          sur l'écran texte
         PLA
         STA Msg+1
         INC Msg
         LDA (Msg)
         INC Msg
         JSR $FB5B
         STZ $24
]loop LDA (Msg)
         BEQ EndTxt
         JSR $FDED
         INC Msg
         BNE *+4
         INC Msg+1
         BRA ]loop
EndTxt LDA Msg+1
         PHA
         LDA Msg
         PHA
         RTS

         DS \

Free = *
```

Légalité

On peut écrire **légalement** des blocs de 524 octets avec l'appel $2030 de GS/OS (cf GS/OS Reference vol 2, $2029 pour la lecture).
Mac Eraser est sous P8 et s'en moque. Sauf sur un point: cela va permettre à Olivier de trouver le "vrai" vecteur ou le "vrai" drapeau.
Rendez-vous dans un prochain ToolBox Mag...

JYB

A cet endroit se trouvent les données relatives aux deux premières pistes d'un disque 800k macintosh. Ces data sont sans intérêt. Vous pouvez les retrouver sur le programme assemblé.

Init ToolsToRam : Table des outils à charger

```
ToolTableDeb  Anop
*   dc.w ToolNum01,MinVersion
*   dc.w ToolNum02,MinVersion
*   dc.w ToolNum03,MinVersion
*   dc.w ToolNum04,MinVersion
*   dc.w ToolNum05,MinVersion
*   dc.w ToolNum06,MinVersion
*   dc.w ToolNum07,MinVersion
*   dc.w ToolNum08,MinVersion
*   dc.w ToolNum09,MinVersion
*   dc.w ToolNum10,MinVersion
*   dc.w ToolNum11,MinVersion
*   dc.w ToolNum12,MinVersion
*   dc.w ToolNum13,MinVersion
    dc.w ToolNum14,MinVersion   WindowManager
    dc.w ToolNum15,MinVersion   MenuManager
    dc.w ToolNum16,MinVersion   ControlManager
*   dc.w ToolNum17,MinVersion   System Loader
    dc.w ToolNum18,MinVersion   QuickDraw Aux
    dc.w ToolNum19,MinVersion   PrintManager
    dc.w ToolNum20,MinVersion   LineEdit
    dc.w ToolNum21,MinVersion   DialogManager
    dc.w ToolNum22,MinVersion   ScrapManager
    dc.w ToolNum23,MinVersion   StandardFile
*   dc.w ToolNum24,MinVersion
;   dc.w ToolNum25,MinVersion   Note Synthetizer
;   dc.w ToolNum26,MinVersion   Note Sequencer
    dc.w ToolNum27,MinVersion   FontManager
    dc.w ToolNum28,MinVersion   ListManager
;   dc.w ToolNum29,MinVersion   Ace
*   dc.w ToolNum30,MinVersion
*   dc.w ToolNum31,MinVersion
;   dc.w ToolNum32,MinVersion   Midi
;   dc.w ToolNum33,MinVersion   Video Overlay
    dc.w ToolNum34,MinVersion   TextEdit
;   dc.w ToolNum35,MinVersion   MidiSynth
;   dc.w ToolNum59,MinVersion   Lisez ToolBox Mag
;   dc.w ToolNum219,MinVersion  Lisez ToolBox Mag
ToolTableEnd  Anop
```

Mise à jour SVP

Lecteurs de ToolBox Mag, lancez le fichier LANCEZ.MOI du catalogue /DIVERS de la disquette et vous saurez si vous êtes à jour. Attention: ***ce n'est pas un poisson !***
Auteurs de ToolBox Mag, vous trouverez au même endroit un petit source Merlin appelé *GetVersion*. Introduisez s'il vous plaît ces lignes au tout début de chacun de vos programmes, que les lecteurs sachent ce qui se passe au lieu d'attendre le plantage.
Notez le truc (certifié par le Bureau Politique): comme les outils ne sont pas là, on envoie au DeathManager par un JumpLong sur son vecteur.

JYB

Et voici le GS Roms 05 !

Patrick Desnoues

Une des caractéristiques les plus attendues du futur GS sera d'avoir l'essentiel de ses outils en Rom: c'est déjà ce qui se passe en partie avec le GS Rom 03, qui vous demande bien moins souvent que son prédécesseur d'introduire la disquette système.
Eh bien, **ce nouveau GS, ToolBox Mag vous l'offre dès aujourd'hui**: avec la petite Init ci-dessous appelée **ToolsToRam**, vous chargez en mémoire tous vos outils qui sont sur le disque système au moment du démarrage, et ensuite ils y restent! **La Ram remplace la Rom**. Cela fait gagner pas mal de temps ensuite au moment du chargement des applications, et vous évite d'avoir à introduire la disquette système à tout bout de champ: les applications ayant besoin de TextEdit, FontManager, StandardFile etc, n'ayant plus à les charger, n'auront évidemment plus à les purger.
L'occupation de la Ram est beaucoup moins importante qu'on ne croit: en fait, il ne vous en coûtera guère que 73k de mémoire. Voilà de toute façon une bonne raison de plus pour ne pas rester juste en mémoire: avec les accessoires et les outils tous en mémoire vive, il ne reste plus guère que les fontes pour demander la disquette système.
Nous pourrions même peut-être, à ce compte, réviser notre position sur la note 53 du Bureau Politique; si tous les outils sont déjà en Ram, on peut peut-être accepter la gymnastique demandée par Apple.

Lors de votre prochaine rencontre avec John Sculley, pensez à lui parler de l'init ToolsToRam de ToolBox Mag: ToolBox étant ouvert à la négociation sur ses droits, **rien n'empêcherait Apple de mettre dans son prochain système une init du même genre** (après mesure de la Ram présente, comme c'est déjà le cas pour ExpressLoad), ou, mieux, de nous permettre de choisir quels outils nous voulons charger en mémoire au prochain redémarrage.

Le programmeur constatera qu'il s'agit d'un programme du type "il suffisait d'y penser". **L'astuce consiste à faire un LoadTools dans une init:** dans une application, le système force la purge des outils en sortie.

Pour l'utilisateur, précisons que l'init en question, appelée **ToolsToRam**, doit être copiée depuis le catalogue /RAMTOOLS.INIT/ de la disquette pour être mise dans le sous-catalogue /SYSTEM/SYSTEM.SETUP de votre disque de démarrage. Vous pourrez choisir de l'activer ou de la désactiver avec le Finder ou Prosel (si vous ne savez pas activer/désactiver un accessoire ou une init avec le Finder, téléphonez à l'Apple II Service Team d'Apple France).

Pour choisir quels outils vous voulez charger au démarrage, il faudra en revanche réassembler le programme (c'est très vite fait, il est très court), en partant de la table ci-contre page 18:
• Les lignes commençant par une '*' correspondent aux outils qui sont déjà en mémoire morte: inutile de les charger...
• Il suffit de mettre un ';' en début de ligne pour que l'outil ne soit pas chargé. C'est ainsi que, dans la version assemblée qui se trouve sur la disquette ToolBox Mag, ne sont pas chargés les outils ToolBox Mag, ni les outils sons, ni le Control Manager (pour tenir compte d'un bug stupide de Wings: si vous n'utilisez pas ce sélecteur, mettez-le), ni l'outil pour la Video Overlay Card.
Il suffit d'enlever ce ';', et l'outil correspondant sera chargé.

[NDLR: ce programme de Patrick Desnoues nous semble susceptible de développements. Un des premiers dont nous serions preneurs est tout simplement un programme Applesoft permettant à son utilisateur de créer sa version de ToolsToRam, en choisissant ses outils dans un menu, sans avoir à réassembler avec APW. Il suffit d'assembler une version avec une table de 255 outils placée en fin de code pour faire un code objet parfaitement "patchable" ensuite sous Prodos 8.]

Comme d'habitude:

LK	: Link
MK	: Génère les macros
Linkall	: Nom des sources à compiler
Macros	: Liste des macros générées par la commande MK
Main	: Programme

← Voir Table des outils Page 18

Test Logiciel : InWords

Le GS sait lire !

Bernard Fournier

Plus simple que sur M...
Plus rapide que sur M...
Plus efficace que sur M...
Plus économique que sur M...

InWords, le logiciel d'OCR du GS.

Attendu depuis plusieurs mois, il est enfin arrivé: le logiciel de **Reconnaissance Optique de Caractères**. Son auteur, Alan Bird, programmeur chez Beagle Bros, a pour la circonstance fondé une autre Société (WestCode SoftWare).
Il est bien entendu indispensable de posséder un scanner: InWords est étudié pour fonctionner avec le scanner **Quickie** de chez Vitesse (il n'est pas impossible qu'il pilote d'autres scanners dans ses versions ultérieures).

Premier contact

InWords est livré sur deux supports: disquette 3'1/2 et disquette 5'1/4, et un manuel conséquent accompagne le programme. Première chose: on branche sa Quickie, on met la disquette InWords dans le mange-disque et on démarre. Deux secondes et trois dixièmes plus tard on est dans le menu principal: pas étonnant ce chargement rapide, car InWords fonctionne en Prodos 8 (mais on peut également lancer le logiciel à partir du Finder).

Le menu principal a un air de déjà vu: il utilise en effet l'interface utilisateur Appleworks, ce qui est somme toute un excellent choix pour un programme 8 bits. Les options proposées permettent d'effectuer un scan de texte, de passer en mode apprentissage ou de changer la police de caractères. Plus simple, tu meurs !

D'abord, deux mots sur la reconnaissance optique: le scanner 'lit' le document posé sur la table et le logiciel 'interprète' ensuite les formes. Pour convertir une succession de points lumineux en lettres, InWords utilise une table de référence: dans cette table (la police de caractères) sont stockées les formes de chaque lettre de l'alphabet (ainsi que les signes, ponctuations, chiffres....). Une succession de points lumineux correspondant à une forme de la table sera traduite en son 'équivalent' ASCII. Simple et efficace, non?

La possibilité de charger une fonte ainsi que le mode apprentissage permettront de définir nos propres tables de caractères selon la typographie des textes que l'on voudra analyser. Le logiciel est fourni avec une dizaine de fontes correspondant aux typographies employées par les principales revues Américaines consacrées à l'Apple. **Il est donc nécessaire de se créer ses propres tables**, ne serait-ce que pour pouvoir reconnaître les minuscules accentuées.

Créer une table de formes

L'apprentissage (la constitution d'une table de référence) est on ne peut plus simple: on règle la Quickie sur la position LETTER, la résolution est fonction de la taille des lettres:

- 400 dpi pour un corps de 10
- 300 " 14
- 200 " 24
- 100 " >24

Enfin on règle la luminosité: pour un premier essai, mettre le repère en face du D de DARK.
Ensuite on passe aux réglages logiciels: indiquer que l'on va créer une fonte, spécifier que l'on va procéder à un apprentissage et régler le degré de correspondance des formes à 70% (faire éventuellement varier ce taux selon la qualité d'impression du texte à analyser: plus le texte est de bonne qualité, plus le taux sera élevé, et plus InWords sera fiable dans ses analyses).
On choisit l'option *Standard Scan* et immédiatement la Quickie s'allume, prête à digitaliser; une fois le texte scanné, un écran Double Haute Résolution affiche une portion du texte digitalisé. Cet écran (que l'on peut déplacer avec les flèches) est l'image graphique du document: cela permet de contrôler si les caractères ont été correctement lus (ajuster la luminosité si besoin et recommencer jusqu'à obtenir des lettres bien formées, pas trop épaisses ni trop 'fil de fer'). Lorsque la digitalisation semble correcte, on appuie sur Return et l'analyse des caractères va s'effectuer. InWords étant en mode apprentissage, il va détecter la première forme parmi le texte digitalisé, va détourer cette forme et demander à quel caractère (ou séquence de caractères) elle doit être associée. On tape la touche correspondante (ou la séquence de touches) et désormais InWords 'reconnaîtra' la forme ana-

lysée comme étant la représentation du caractère (ou des caractères) qu'on a indiqué. Lorsque l'apprentissage est terminé, on peut sauver la fonte pour pouvoir en disposer ultérieurement (voire pour la compléter).
Remarque: les minuscules accentuées utilisant l'accent circonflexe ou le tréma devant être traduites en caractères ASCII <128, le programmeur a utilisé la méthode suivante: on peut spécifier à InWord qu'une forme devra être traduite en une suite de plusieurs caractères ASCII. Par exemple:

â	->	^a
ë	->	¨e
©	->	(c)

Avec ça, aucun problème pour analyser tous les textes possibles, même ceux mélangeant caractères US et minuscules européennes: tout est question d'apprentissage pour InWords.

Dans l'option *Fonte* du menu principal, on a la possibilité d'éditer les caractères de la table: cette option permet en réalité de visionner le nombre de formes différentes associées à un même caractère ASCII (on peut également supprimer une ou plusieurs formes en cas d'erreur). En effet, si un texte possède des variantes dans les formes d'une même lettre, InWord va enregistrer toutes ces variantes comme étant équivalentes à un seul caractère ASCII. Exemple:

a a *a* ***a*** seront associés au caractère a,
et dans la table des formes, en face de 'a' sera indiqué 4, pour rappeler que 4 formes différentes sont associées au caractère 'a'.
Une fois encore, tout est question d'apprentissage. **InWords peut faire tout ce qu'on veut, à condition de le lui avoir appris** !

Les fonctions avancées

On dispose de trois modes de scan: *standard*, *colonne* et *lier*. Le premier mode correspond au scan d'un texte de moins de 10 cm de large (cette largeur est celle du scanner; la longueur étant fonction de la mémoire disponible). Les deux autres modes sont nettement plus intéressants: le mode *colonne* va permettre de distinguer un texte formatté en colonnes; et le mode *lier* va permettre de scanner un texte disposé en lignes de moins de 20 cm.

Mode Colonnes: imaginons un texte disposé de la manière suivante:

```
xxxx x xxx xxxxx xx      yy yyy yyyy yyyyyy y
xxxxxxx x xxxx xxx       yyyy y yy yyyy y yyyy
```

Ce texte est disposé en colonnes; le scan se faisant sur une largeur de 10 cm, en mode *standard* chaque ligne serait analysée avec une partie de la colonne 2. En employant le mode *colonne*, on indique à InWords que les espaces en fin de ligne sont à considérer comme la fin de la zone à analyser. On va donc pouvoir scanner les colonnes les unes après les autres, le texte sera analysé correctement.

Mode Lier: avec un texte ayant une largeur supérieure à 10 cm (et inférieure à 20 cm), ce mode permet de scanner la partie gauche **puis** la partie droite (sans se soucier d'un éventuel chevauchement): InWords détectera les chevauchements et les corrigera en conséquence.

Les textes analysés sont sauvés soit en format Texte (CR en fin de ligne), soit en format AppleWorks-8, ce qui permet leur exportation dans tous les traitements de texte du GS.

Les limites d'InWords

• InWords est extrêmement sensible au réglage de la luminosité de la Quickie, cela signifie également qu'il ne peut reconnaître que du texte bien contrasté (noir sur fond blanc).
• On ne peut pas scanner un texte comportant des zones graphiques (il manque la possibilité d'indiquer au logiciel que certaines zones sont à occulter): il reste la méthode du masquage avec des PostIt !
• Le logiciel fonctionne avec des lettres de taille 10 minimum: avec des tailles inférieures, il ne peut distinguer les formes et fait des confusions. De même, l'espacement entre les lettres doit être suffisant pour qu'il puisse les distinguer les unes des autres, sinon bonjour les confusions entre 'ri' et 'n' ou entre 'nn' et 'm'. Pour pallier cet inconvénient, on peut toujours agrandir le texte avec une photocopieuse qui permet l'agrandissement.
• Les lettres doivent être bien formées (qualité laser, photo-gravure, offset) ce qui exclut les listings produits par une imprimante matricielle, les textes des magazines bon marché, et bien évidemment les textes manuscrits.
• Enfin les confusions inévitables pour ce type de produit: les 0 (chiffres) et O (lettres), les I (i majuscule) et les 1 (le chiffre), et dans une moindre mesure (tout dépend de la taille des lettres) les O, D et Q.
• Et puis malheureusement, il faut signaler les trop nombreux plantages du logiciel: défaut de jeunesse certes, mais défaut tout de même... heureusement qu'il suffit de faire Control-Reset pour revenir dans InWords sans perdre le document en mémoire; mais quand même cela fait désordre !

Malgré ces quelques restrictions, à l'usage on constate qu'InWords reconnaît les textes avec un pourcentage d'erreur inférieur à 2%. **Les phases d'apprentissage doivent requérir toute l'attention et le soin de l'opérateur**: de la qualité de son travail dépend la fiabilité de ses reconnaissances optiques futures.

Un GS grand écran :

Border Scroll

Dominique Delay

Mais non, les bordures de votre écran ne sont pas des zones vides, neutres, sur lesquelles on ne peut rien afficher: "Border Scroll" va vous montrer comment afficher et faire défiler du texte dans la bordure inférieure de l'écran!

L'utilité d'une telle routine n'est pas évidente, (encore que l'on puisse par exemple imaginer un programme de dessin plein écran avec des options dans la bordure): prenons donc sa réalisation comme un défi intellectuel et une occasion de discuter certains aspects du GS dont on parle peu.

Pour l'utilisateur

Passez sous Basic System. Quand vous voyez le "prompt" du Basic, tapez franchement Control-Reset (pas Ó-Control-Reset, qui vous ferait rebooter, mais un Reset simple). Il est en effet possible, si vous êtes sous GS/OS, que d'autres que "Border Scroll" envoient des interruptions, cherchent à se synchroniser sur les VBL, etc, et empêchent par là Border Scroll de rester synchronisé. Tapez ensuite:

BRUN /TOOLBOX.MAG.04/BORDER.
SCROLL/BORDER.

La fenêtre centrale de votre écran va alors passer en bleu. Mais **regardez bien ce qui se passe en-dessous**: vous voyez du texte défiler dans la bordure! Vous pouvez interrompre et reprendre le défilement à tout instant en tapant Ó-Option-Control-Shift-Delete. Pour quitter Border Scroll, un Control-Reset suffira.

Pour le bidouilleur

Quel que soit le mode graphique dans lequel se trouve le GS, la bordure est toujours présente. Il est intéressant de noter que sa position peut varier quelque peu: ainsi la limite inférieure écran-bordure est plus basse en Super Haute Résolution qu'en mode texte. En théorie, la bordure est de couleur uniforme, choisie parmi les 16 disponibles.

L'astuce que nous allons utiliser consiste à **changer la couleur de la bordure le plus souvent possible, tout en gardant le 65816 parfaitement synchronisé avec le VGC** (processeur graphique). L'animation synchronisée sera exécutée en vitesse lente, à 1 Mhz.

Avant toute chose, livrons-nous à quelques calculs (pour lequels nous supposerons le GS en vitesse lente) :

A 1 Mhz, nous disposons d'environ un million (10^6) de cycles par seconde. En une seconde le processeur vidéo balaye l'écran 50 fois (en 50 HZ), nous disposons donc d'environ 10^6/50 = 20 000 cycles de microprocesseur par balayage.

Maintenant, un écran complet (bordures comprises) en 50 HZ correspond à 312 lignes balayées. Le nombre de cycles correspondant au balayage d'une seule ligne est alors : 20 000/312 = 64.10 . En pratique, on verra que ce nombre est en fait de 65.

Afin d'obtenir la meilleure 'résolution' possible, il faut changer la couleur de la bordure le plus souvent possible: ceci sera accompli en utilisant les formes courtes (page zéro) des instructions de stockage, ce qui présuppose que l'on positionne la page zéro au bon endroit!

L'astuce consiste à assigner la valeur $C000 au registre D de page directe (mettre la page zéro en $C000), ce qui permet d'adresser le registre de la couleur d'écran ($C034) par de simples

Références

Qu'est-ce que le VGC? Le registre $C02F? Quest-ce qu'un cycle du microprocesseur? Une seule référence: le livre **"Le II GS Epluché"**. La partie "graphisme et animations" de ce livre a pour auteur... Dominique Delay.

Dominique est aussi l'auteur, vous l'avez noté, de l'animation de présentation de la disquette ToolBox Mag 4. N'oubliez pas de booter la disquette !

JYB

STX $34. Cette instruction prenant 3 cycles, nous aurons une 'résolution' de 65/3 = 21 éléments horizontaux.
Arbitrairement, la largeur de nos caractères sera de 3 éléments, 4 avec l'espace qui les suit, nous permettant ainsi d'afficher 5 caractères simultanément à l'écran. Leur hauteur sera de 14 lignes, avec une ligne sur deux laissée noire, simple question d'esthétique.

Arrivés à ce stade, nous savons que, une fois synchronisés, nous n'avons plus qu'à changer la couleur de la bordure tous les 3 cycles, avec un modulo de 65 cycles, pour obtenir une figure stable à l'écran. La question qui subsiste est bien entendu comment se synchroniser ?

Comment se synchroniser ?

En fait, la méthode utilisée est très empirique. Le balayage vidéo se faisant "en Z" (balayage d'une ligne, retour à zéro, descente d'une ligne), il est facile de se synchroniser verticalement: il suffit d'attendre une valeur particulière du registre VERTCNT ($C02E).
En revanche, la synchronisation horizontale est une autre affaire. Le registre HORIZCNT ($C02F), reflète bien à tout instant la position horizontale du faisceau, mais sa valeur change bien trop rapidement pour que l'on puisse à coup sûr se synchroniser sur une valeur particulière (un triplet LDA CMP BNE prend au minimum 7 cycles, soit 1/9 du temps de balayage d'une ligne!). Si seule la synchronisation horizontale importait, on pourrait attendre que HORIZCNT ait une valeur particulière, mais dans notre exemple, on désire être synchronisé à la fois verticalement et horizontalement !

Dans ce cas, une seule solution subsiste : se synchroniser verticalement en attendant une valeur particulière de $C02E, puis lire $C02F et exécuter une routine de temporisation différente pour chaque valeur possible lue à ce moment précis. Le nombre de valeurs différentes que l'on peut lire est (toujours empiriquement) de 8, elles dépendent de la désynchronisation qui existe à cet instant précis entre le VGC et le 65816, le VGC 'travaillant' plus vite que le processeur.
Après le passage au travers de la routine (composée d'une série de CMP, branchant à différents niveaux d'une série de NOPs et autres BITs, prenant respectivement 2 et 3 cycles), le VGC et le 65816 sont parfaitement synchronisés. En pratique, cela signifie qu'un LDA $C02F suivant directement cette routine donnera toujours la même valeur.

Défilement du texte

Une fois la synchronisation obtenue, un certain nombre de routines composées de 21 STZ/STX courts + un NOP, totalisant ainsi 3*21+2 = 65 cycles, sont exécutées. Le registre X est au préalable chargé avec une valeur représentant une couleur de bordure: chaque STX correspond à l'affichage d'un élément plein, alors que chaque STZ (STore Zero) correspond au passage de la bordure à la couleur noire.

Pour faire défiler ce texte, nous pourrions simplement bouger les instructions STX/STZ dans la routine elle-même, mais notre 'résolution' de mouvement ne serait que de 1/21e de l'écran environ ! Encore pire qu'un scroll en mode texte !
Pour améliorer le pas de défilement, on peut agir directement sur la routine de synchronisation, en rajoutant ou en enlevant un cycle à la fin de cette dernière. Un cycle correspond à 1/65 d'une ligne, le pas le plus fin que l'on puisse avoir.

On est de toute façon obligé de bouger les instructions au cœur de la routine (correspondant à un pas de 3 cycles), mais pour que le mouvement soit le plus fin possible, on procède comme suit : on a trois routines de temporisation post synchronisation, chacune différant de la précédente d'un seul cycle. Au premier balayage, on exécute la première routine, au deuxième balayage la deuxième, au troisième balayage la troisième, puis, lors du quatrième balayage, on bouge les instructions à l'intérieur du programme tout en recommençant à exécuter la première routine de temporisation... et ainsi de suite.

Notes finales

Le reste de la routine est assez classique. Il est encore intéressant de noter que l'on peut facilement rajouter un dégradé de couleurs dans les caractères, en transformant les NOPs coincés entre les instructions de stockage en LDX, changeant ainsi la couleur à chaque ligne.

Si l'on désire afficher une page SHGR en même temps que le scroll, il vaut mieux repasser en mode texte (C029:41) avant de se synchroniser, sous peine d'avoir des 'parasites' autour des barres....

Vous trouverez le source Merlin dans le sous-catalogue /Border.Scroll/ de la disquette ToolBox Mag 4.

Le GS pour débutants
première partie : avant de démarrer

Eric Weyland

A lire votre courrier, à vous répondre au téléphone, on se dit que les ventes d'Apple II GS ont dû connaître une gigantesque explosion ces derniers temps: ***c'est fou le nombre de "débutants" qu'on trouve sur cette machine !***

Vous qui êtes perdu dans la jungle des Inits, CDevs, Drivers, ProDos, P8, GS/OS, CDA, NDA, Fontes, Tools, PIF, TIF, S16, SYS, Icône, Control Panel, vous qui appelez ou écrivez à ToolBox Mag pour demander comment on lance un accessoire de bureau, ou pourquoi vous n'arrivez pas à lire vos disquettes 5,25, cette série d'articles vous est destinée: après sa lecture (qui suppose que vous avez déjà lu les documentations remises avec le GS et avec son système), vous n'aurez plus l'excuse d'être un débutant. Si vous trouvez ces explications trop difficiles, une seule solution: téléphoner à l'Apple II Service Team d'Apple France.

Vous qui n'êtes plus débutant, jetez-y quand même un coup d'œil: je tâcherai de faire cette initiation de façon à passer tout de suite à une pratique "évoluée" du GS, en raccourcissant le chemin que nous avons tous dû prendre pour savoir gérer la Bram ou charger des accessoires.

Avez-vous vraiment un Apple II GS ?

Tout d'abord, assurez-vous que vous possédez **un véritable Apple IIGS**. Voici le minimum que vous devez avoir:

• Un **moniteur** (un écran) relié à l'ordinateur: couleur ou monochrome, cela n'a pas d'importance ici, sauf pour **certains moniteurs couleur à prise Péritel**, reliés par un câble GS/Péritel: beaucoup, datant de l'Apple IIe, ont une **résolution insuffisante** pour faire un affichage propre en mode graphique 640 du GS. Si c'est le cas du vôtre, deux solutions: soit vous le bazardez pour acheter un moniteur couleur Apple, soit vous achetez un moniteur monochrome de IIe ou de IIc d'occasion, que vous brancherez **en plus** de l'autre.

• Une **unité centrale**, qui est l'ordinateur à proprement parler. Elle consiste en un boîtier gris sur lequel est écrit "Apple II GS". [**NB**: si sur votre boîtier, il y a en plus une espèce de gribouillis bizarre où on peut lire quelque chose comme "Woz...", apportez-le **d'urgence** à ToolBox pour un **échange entièrement gratuit**.]

Attention: ce qui est dans cette boîte doit être un ordinateur Apple IIGS **Roms 01 ou 03**, et ce n'est pas toujours le cas. Pour déterminer le numéro de version de votre Rom, il suffit de mettre l'ordinateur en marche et de lire le texte qui s'inscrit avant le démarrage du lecteur de disquettes. Le numéro de version de Rom de votre Apple IIGS est affiché en bas de l'écran. Sur un Apple IIGS Rom 01, cela donne :

Apple IIgs

Copyright Apple Computer, Inc. 1977-1987
All rights reserved.
ROM VERSION 01

Le numéro de la Rom apparaît en clair sur la dernière ligne. Sur un Apple IIGS Rom 03, qui est la machine vendue à ce jour, le texte est identique à part que la mention ROM VERSION 01 est remplacée par ROM VERSION 03.

S'il n'y a pas de numéro de version de Rom, vous avez devant vous une pièce de musée : un Apple IIGS Rom version 0. Prenez rapidement rendez-vous, non pas avec un antiquaire, mais avec un revendeur Apple pour effectuer une mise à jour du précieux circuit. Ce dernier va se faire un plaisir de vous remplacer votre carte mère contre une nouvelle ayant une Rom 01 (il n'y a pas de possibilité de mise à jour entre les Roms 01 et 03, sauf en faisant l'acquisition pure et simple d'un Apple IIGS). Si le concessionnaire Apple ne sait pas faire ce travail (ultra-simple), demandez à Apple France la liste de ses revendeurs à même de le faire. En désespoir de cause, ToolBox pourra vous sortir de cette

ornière.

• Un **lecteur de disquettes 3,5**: retournez-le et vérifiez qu'il est bien écrit dessous "Apple 3,5 Drive". S'il y a écrit "Unidisk 3,5", vous allez avoir beaucoup de problèmes, et nous vous conseillons d'acheter un Apple 3,5.

• un **clavier** détachable et une **souris**: attention, **la souris doit être connectée sur le clavier.** Si la vôtre se connecte sur une carte à l'intérieur du GS, un très grand nombre de programmes GS ne la reconnaîtront pas.

• **Assurez-vous que vous avez de la mémoire en quantité suffisante.** A l'heure actuelle, **le minimum absolu est de 1 Mégaoctet** de mémoire vive, mais **deux Mégas sont quasiment indispensables.**
• sur les GS Rom 01, il doit y avoir une carte d'extension mémoire dans le slot mémoire de l'ordinateur. Il s'agit en général de la carte d'extension mémoire Apple, qui au maximum est extensible à 1 Mo par l'adjonction de puces de 256 kilobits; complète, cette carte possède 32 puces mémoire. Dans le cas contraire, vous n'avez **pas assez de mémoire**.
Je vous déconseille de compléter cette carte. Passez tout de suite au stade supérieur en faisant l'acquisition d'une carte d'extension mémoire ayant une capacité d'adressage plus importante (4 Mégas par exemple), et mettez-y au moins deux mégas de mémoire.
• sur les GS Rom 03, il y a déjà 1 Mo de mémoire soudé sur la carte mère. Le slot d'extension mémoire est, dans la configuration de base, dépourvu de toute carte. **Il y a là un trou à combler d'urgence !** Vous pouvez faire l'acquisition d'une carte mémoire extensible à 4 Mo, ou trouver d'occasion une carte d'extension mémoire Apple à 1 Mo (de nombreux nababs ayant muni leur GS de plusieurs mégas de Ram ont ce genre d'article dans leur placard...). Votre GS aura dès lors deux Mégas de mémoire.

A partir de maintenant, je considère donc que vous tournez sur l'ordinateur dont il est question dans ToolBox Mag: **un véritable Apple IIGS**, et non pas un ZX81, ni un compatible Apple II dans un Macintosh à bas prix.

Pour être franc, il y manque encore une chose: **un disque dur (ou un lecteur de cartouches genre Syquest) de 20 Mégas ou plus**. Disons que ce genre de périphérique n'est pas indispensable pour **apprendre** à se servir d'un GS, mais seulement pour **s'en servir effectivement** une fois qu'on sait. Cela pourra donc faire l'objet d'un article ultérieur.

Ce qu'il faut jeter

Attention ! Pas si vite ! Il vous faut aussi **un système d'exploitation à jour.** Mettez immédiatement à la poubelle les disquettes Dos 3.2, Dos 3.3, ProDos 8, ProDos 16, MouseDesk, Système 3.1, et Système GS/OS 4.0 (dans le lot, je vous conseille de vous débarrasser aussi de GS/WRITE). **Le système à jour est le GS/OS 5.04.** Il est en français, accompagné d'un manuel d'utilisation en français. Ne vous étonnez pas que le manuel parle de GS/OS 5.02 au lieu de 5.04: voyez l'article de J.Y. Bourdin dans ce numéro. Avant de continuer, je vous invite à lire attentivement le manuel en question. Il vous dira tout sur les manipulations de base.

Maintenant que tous ces éléments sont réunis, vous avez **un** Apple IIGS. Ne croyez pas que vous allez pouvoir tout de suite lancer une disquette: il faut d'abord faire de cet Apple IIGS anonyme **votre** GS. Il faut le **configurer**.

Le tableau de bord du GS

Pour configurer votre GS, il faut passer par le Tableau de bord (ou Control Panel). Il existe trois versions de ce tableau de bord: la première est en anglais et en mode texte, et se trouve dans la mémoire morte (Rom) du GS. C'est celle dont nous allons parler aujourd'hui. La seconde est la même que la première, mais en français. La troisième est un autre tableau de bord, graphique: la seconde et la troisième étant apportées par les disquettes système, nous en parlerons une prochaine fois.
Le rôle de ce tableau de bord est de pré-régler un certain nombre de paramètres. Le GS gardera en permanence en mémoire, même une fois éteint, ces paramètres dans une mémoire conservée avec une pile, appelée "BRam" (Battery Ram).
Cette pile a une durée de vie limitée : si par exemple votre GS se met à se comporter étrangement (affichage en 40 colonnes, réglages des couleurs autres que ceux que vous avez choisis, affichage en caractères américains, clavier en QWERTY, pendule pas à la bonne date...), ne cherchez pas plus longtemps la cause du mal (ce n'est pas un virus) : **la pile est à changer**.
C'est peu de chose à faire, et le service technique de tout concessionnaire Apple est capable de faire cette intervention; attention quand même à ne pas vous faire "arnaquer" par quelqu'un qui vous imposerait de changer la carte-mère, ce qui revient à changer la voiture quand le réservoir

est vide.
Certains programmes, sans prévenir, vont modifier vos réglages dans le Control Panel pour des raisons de commodités de fonctionnement ; le problème est qu'ils oublient souvent de remettre les lieux dans l'état dans lequel ils les ont trouvés en entrant. Pour pallier ce genre de désagrément, vous pouvez utiliser un programme qui sauvegarde la BRam sur disque pour pouvoir la restaurer en cas de besoin. Pour cela, vous trouverez sur la disquette ToolBox Mag 4, dans le sous-catalogue /DIVERS, un petit utilitaire en français, *Bram.Util* (écrit par Jean Luc Darbon), qui permet de faire ce travail. Il vous donne aussi la possibilité de visualiser et de nettoyer le contenu de votre BRam des pollutions laissées par ces programmes sans gêne.

Pour accéder au tableau de bord dit "en CDA" (en mode texte), il suffit de taper ⇧+ Control + Esc., puis de sélectionner *Control Panel* (qui doit être le premier titre du menu). Nous allons maintenant procéder aux réglages internes de votre Apple IIGS en expliquant à quoi sert chacun d'entre-eux. Vous devez avoir sur votre écran, et donc sous vos yeux, la liste suivante :

Display
Sound
System Speed
Clock
Options
Slots
Printer Port
Modem Port
Ram Disk
Quit

L'heure et la date sont affichées en haut à droite de l'écran dans un rectangle.
Dans tous les réglages du Control Panel, le petit repère à gauche d'une option indique qu'il s'agit d'un réglage par défaut.

• L'option *Display* (Affichage) permet d'indiquer à l'Apple IIGS le type de moniteur qui y est relié : monochrome ou couleur ; le nombre de colonnes d'affichage en mode texte : 40 ou 80 (en général, on choisit 80) ; puis les couleurs d'affichage du texte, du fond et de la bordure (vous êtes libre de vos choix). Le réglage *Standards* permet de remettre rapidement les couleurs de réglages d'origine pour le texte, le fond et la bordure.
Certains virus s'ingénient à changer ces réglages en réglant des couleurs impossibles à obtenir manuellement (par exemple affichage de texte noir sur un fond noir). Dans ce cas, inutile d'emmener votre GS chez le concessionnaire du coin pour un échange standard de la carte mère (cela est arrivé trop souvent): il suffit de remettre en place les réglages par défaut. Pour cela, tapez (en aveugle si un virus de ce genre vous a contaminé) Option + Control + Reset. Il y a alors quatre choix qui se présentent :

1 = *Enter the Control Panel* (accéder au Tableau de bord)
2 = *Set system standard and 60 hertz* (réglages par défaut et 60 hertz)
3 = *Set system standard and 50 hertz* (réglages par défaut et 50 hertz)
4 = *Continue restarting the system* (redémarrage de l'ordinateur)

Tapez 2 ou 3 selon la fréquence utilisée ; tout redevient normal...

La fréquence actuelle de votre système est indiquée en bas de l'écran quand on sélectionne l'option *Display* du Control Panel. Le système TV américain diffère du système européen par sa fréquence de rafraîchissement d'image. Celle-ci est de 60 hertz aux Etats-Unis pour 50 hertz en Europe. 60 hertz signifie simplement que le processeur vidéo rafraîchit l'écran 60 fois par seconde ; qu'il envoie 60 fois par seconde les données d'affichage de chaque ligne vers l'écran. La plupart des utilisateurs de GS, pour des raisons diverses, préfèrent un réglage en 60 hertz. Choisissez ce qui convient le mieux à votre moniteur.

• L'option *Sound* (Son) permet de régler le volume sonore et la fréquence du son de l'Apple IIGS.

• L'option *System Speed* (Vitesse du système) permet de régler la vitesse de fonctionnement de l'Apple IIGS : rapide 2.8 Mhz ou normale 1 Mhz. La plupart du temps on reste bien évidemment en vitesse rapide. La vitesse lente est conseillée pour utiliser certains logiciels développés sur Apple II, qui ont des routines de calcul et de temporisation pointues, qui ne peuvent pas être accélérées. C'est souvent le cas avec les logiciels protégés, les programmes de copie et les logiciels de communication. Certains jeux d'arcade de l'ancien temps sont plus amusants en vitesse rapide. On peut aussi passer en vitesse lente pour tricher à certains jeux GS ; *essayez par exemple avec le jeu d'Olivier Goguel de ce numéro* ; cela permet de voir venir...

• L'option *Clock* (Horloge) permet de régler l'heure et la date du GS, ainsi que de définir les différents formats d'affichage.

• L'option *Options*permet de réaliser un certain nombre de réglages, dont les plus importants sont le jeu de caractères d'affichage (français ou autre), le type de clavier (AZERTY ou autres), le réglage de sensibilité de la souris, le buffer (tampon) clavier (certains programmes plantent systématiquement quand ce réglage est actif, il est donc conseillé de le mettre sur Non) et le réglage de la sensibilité des touches du clavier.

• L'option *Slots* (Connecteurs) est vitale à comprendre. En effet, le IIGS est un ordinateur qui est muni de slots (les connecteurs internes) afin d'y loger des cartes d'extension (TransWarp, Quickie, Carte SCSI...), et des ports (des sorties) permettant de brancher directement (sans carte d'extension) divers périphériques (imprimante, modem, lecteurs de disquettes...). Il faut absolument comprendre qu'électroniquement **chaque port est relié à un slot**, et qu'il est en pratique difficile d'utiliser ensemble un port et un slot électroniquement connectés ; évidemment il y a des exceptions qui permettent de cumuler port et slot. Voici la liste des connexions entre ports et slots pour un Apple II GS Rom 01:

Port imprimante = Slot 1
Port modem = Slot 2
Port vidéo = Slot 3
Port souris = Slot 4
SmartPort (lecteur 3.5") = Slot 5
SmartPort (lecteur 5.25") = Slot 6
AppleTalk = Slot 7

Le réglage entre port et slot se fait à partir de l'option *Slots* du Control Panel. C'est là que l'on indique à l'Apple IIGS si on utilise une carte d'extension dans un slot (dans ce cas, on choisit comme réglage *Your Card*) ou le port intégré qui lui correspond. *Les deux ne peuvent pas être actifs en même temps.* Ainsi, si on branche une imprimante ImageWriter sur le port imprimante, en slot 1 il faut configurer *Printer Port* (nous verrons une prochaine fois qu'il faut aussi un programme pour gérer cette imprimante : cela s'appelle un driver) ; il est toujours possible de mettre une carte d'extension en slot 1 (par exemple une carte d'acquisition de données), mais elle ne sera pas reconnue par le système. Il existe cependant des cartes intelligentes qui font automatiquement la commutation entre le port et le slot. C'est le cas de la carte accélératrice TransWarp GS qui placée en slot 3 ou 4 ne nécessite pas un réglage sur *Your Card* (votre carte) à la place de *Built-in text display* (affichage texte intégré) ou *Mouse Port* (Port Souris). La plupart des cartes stéréo fonctionnent de façon analogue. La carte d'interface du scanner à main Quickie est de la même façon une carte intelligente.

Pour les lecteurs de disquettes, c'est un peu plus compliqué. En effet, il est possible de constituer **une chaîne de lecteurs** : lecteurs 3.5" et lecteurs 5.25" ; le tout étant connecté sur le SmartPort (le port disque). Pour avoir accès aux lecteurs 3.5", il faut donc indiquer *SmartPort* en *slot 5* (+ un driver dans le système) ; de même pour les lecteurs 5.25", il faut indiquer *DiskPort* en *slot 6* (+ un driver dans le système). C'est sous ces deux conditions (réglages dans le Tableau de bord puis installation d'un driver adéquat dans le système) que votre lecteur sera pris en compte sous GS/OS. Nous verrons une autre fois comment on installe un driver dans le système.

La situation peut être plus compliquée si vous avez une carte d'interface ou un disque dur interne sur carte (l'Insyder de chez Cirtech par exemple) en slot 6. Dans ce cas, il sera impossible de faire des copies de fichiers entre le lecteur de disquettes 5.25" et le disque dur, parce que le premier est placé dans un slot et que le second est branché sur un port qui lui est électroniquement semblable.

C'est également à ce niveau du Control Panel que l'on choisit le **slot de démarrage**. Il faut bien entendu que ce numéro de slot corresponde à un périphérique disque. La solution la plus simple est de régler le *Startup* (Démarrage) sur *Scan* (recherche). Dans ce cas, c'est le premier périphérique trouvé qui démarrera (la recherche commence par le slot 7, puis 6, etc). Sinon on peut, pour gagner quelques secondes, indiquer directement le numéro de slot sur lequel on veut démarrer. Il est également possible de démarrer sur un disque Rom ou Ram.

• Les options *Printer Port* et *Modem Port* (Port imprimante et Port modem) permettent de spécifier les caractéristiques de communication (protocoles) entre l'imprimante et/ou le modem et l'Apple IIGS. Pour les périphériques Apple les réglages par défaut conviennent à merveille. Pour les périphériques non Apple, il est conseillé de lire les notices techniques d'utilisation et de procéder à tâtons pour trouver les réglages corrects.

• Enfin, l'option *RAM Disk* (Disque RAM) permet de fabriquer un disque virtuel dans l'Apple IIGS. A toutes fins utiles, je rappelle qu'un disque Ram est une partie de la mémoire adressable de l'Apple IIGS que l'on peut réserver afin qu'elle se comporte comme un lecteur de disquettes. Les temps d'accès sont bien en-

tendu considérablement accélérés. La partie de cette mémoire se comporte alors comme n'importe quel volume disque ; c'est à dire qu'elle a un catalogue et que l'on peut y copier des fichiers et des programmes : ils y resteront jusqu'à l'extinction de la machine.
Attention, si vous voulez pouvoir booter (démarrer) sur un disque Ram ou un disque Rom, c'est possible. Mais il ne suffit pas de les déclarer dans le Control Panel, **il faut encore les formater avec le Finder.**

Quand on déclare un disque Ram, on peut lui allouer une taille. Cette taille, en nombre de Ko, est prise sur la mémoire totale disponible dont dispose votre Apple IIGS. **Si vous ne possédez que 1,2 Mo de mémoire, vous ne devez pas déclarer de Disque Ram**. Votre configuration est beaucoup trop pauvre pour tourner sous GS/OS et accepter ce luxe.
La présence d'un Disque Ram peut aussi expliquer certains plantages ; de nombreux logiciels ont besoin de mémoire pour fonctionner et plantent piteusement s'ils ne la trouvent pas.
Si vous avez un Disque Ram, il sera électroniquement équivalent au slot 5, drive 2. **Votre deuxième lecteur de disquettes 3.5" sera alors déplacé en slot 2, drive 1** ; ce qui provoquera une jolie pagaille chez les personnes non averties. Dans tous les cas, il est conseillé de **régler la taille minimum du disque Ram sur la même valeur que la taille maximum**. Le contenu du Disque Ram n'est pas affecté par un ⌘+ Control + Reset, c'est pourquoi **vous pouvez rebooter (redémarrer) sur le disque Ram** (qui sera alors reconnu en slot 5 drive 1). Pour vider le contenu du disque Ram, il faut taper ⌘+ Control + Shift + Reset (vous avez **dix** doigts, non?).

Dans la plupart des cas où vous modifiez vos réglages, il est conseillé de rebooter (redémarrer) pour que ces réglages soient pris en compte par le GS (surtout les réglages concernant les slots).

Et voilà: vous avez enfin un vrai GS, tout prêt à fonctionner. **Vous allez enfin pouvoir démarrer votre première disquette**. La seule disquette que vous devriez **jamais** démarrer, d'ailleurs: la disquette système.
Pour nous, nous en resterons là pour cette fois-ci. D'ici le prochain ToolBox Mag, vous pourrez vous avancer en lisant la documentation de GS/OS 5.04 et en utilisant ce système. La deuxième partie de l'initiation à l'Apple II GS sera en effet consacrée à une **anatomie des deux disquettes système 5.04.**

Egalité

Avec la baisse de prix des nouveaux Macs, qui n'avait pas été accompagnée d'une baisse équivalente sur le GS, avec HyperCard livré gratuitement avec les Macs, mais pas avec les GS, avec comme seul nouveau modèle d'Apple II une pièce de musée à mettre dans un Macintosh à bas prix, les utilisateurs d'Apple II GS avaient quelque raison d'être jaloux et d'avoir des aigreurs à l'égard d'Apple. Nos colonnes s'en sont fait l'écho.

Nous sommes donc d'autant mieux placés pour **dire de tout cœur "bravo" à Apple France** pour la double décision commerciale qu'il vient de prendre en faveur du GS:

❶ Tous les Apple II GS neufs vendus à partir du 01/04/91 seront munis de **deux Mégas de mémoire** (une carte mémoire Apple 1 Méga remplissant le slot d'extension mémoire), et cela pour le même prix qu'avant. C'est une nécessité, car les nouveaux GS seront en outre **livrés avec HyperCard en français**, et le système 5.1 en français.
Enfin, précisons: ils seront livrés avec le 5.02 et avec **un bon à remplir pour recevoir ultérieurement HyperCard et le système 5.1**. Là, vous ne pouvez pas rouspéter: si vous achetiez un Mac, ce serait la même chose. Hypercard GS, ce sont trois manuels et six disquettes. Il faut du temps, c'est normal, à la bureaucratie Apple pour faire la version française. L'essentiel est bien l'égalité désormais rétablie entre GS et Mac.

❷ Apple connaît son monde, et a prévu les cris des propriétaires actuels de GS: **tous les "utilisateurs enregistrés" d'Apple II GS peuvent aller retirer gratuitement le bon ci-dessus chez le concessionnaire Apple le plus proche**. Qu'appelle-t-on "utilisateur enregistré" de GS? C'est simple: c'est soit un acquéreur du système 5.02 français, soit un abonné au "Guide de l'Apple II" d'Apple France.
Concrètement, il suffit de vous rendre chez votre concessionnaire avant le 01/05/91 avec sous le bras la Documentation du 5.02 ou le Guide de l'Apple II de Février 91, de lui demander le "bon de mise à jour Apple II GS", de remplir et de renvoyer aussitôt ce bon à l'adresse de l'Apple II Service Team.
Les concessionnaires Apple étant ce qu'ils sont, il peut être utile d'insister en leur donnant la "Référence Produit" de la mise à jour Apple II GS: *506F6973736F6E*. Sinon, essayez un des concessionnaires de la liste donnée dans le Guide de l'Apple II. En désespoir de cause, téléphoner à Apple France.

Quoi qu'il en soit, même s'il est **simplement normal**, dans le fond, de mettre ainsi le GS sur un pied d'égalité avec le Mac:

Bravo Apple France !

Le monde à l'endroit
Un patch à Wings

J.Y. Bourdin

Dans notre numéro 3, vous avez pu lire (page 50) de véhémentes protestations contre le "monde à l'envers", l'esclavage à la Mac auquel voudrait nous soumettre le Wings de Vitesse, qui n'a pas de vrai Quit comme tout le monde.

Rouspéter ne suffit pas à nous délivrer de l'esclavage: voici donc **un patch pour donner un Quit normal à Wings**, et montrer à Vitesse que ses tentatives sont inutiles. Ce patch permet de mettre Wings où on veut sur ses disques, avec le nom qu'on veut, et pas forcément en /System/Start, et de le lancer et le quitter comme on veut: bref, il en fait une application normale.

A noter: les chaînes hexa ci-dessous sont valables pour la version 1.3 de Wings, et se font sur le fichier principal de Wings, le fichier /System/Start de la disquette Wings.

Patch simple

Le patch le plus simple permet que Wings ne se mette plus dans la pile des retours quand il lance Basic System depuis son bouton "BASIC" préprogrammé. Après avoir cliqué sur ce bouton BASIC sous Wings, et être passé sous Basic System, tapez "Bye": vous ne retournez plus à Wings, mais à votre lanceur précédent. Si vous avez mis Wings en /System/Start, il sera relancé: mais ce patch n'a de sens que pour ceux qui veulent mettre Wings ailleurs qu'en /System/Start.
Patch: remplacer
[AD FA 7E 8D FC 86 22 A8 00 E1 29 20] par
[AD FA 7E 9C FC 86 22 A8 00 E1 29 20]

Pour les fainéants qui ne veulent même pas taper "Bye" et veulent simplement cliquer sur Quit, voici un autre patch, plus complet.

Patch complet

Le patch suivant permet de transformer le bouton préprogrammé "Finder" de Wings en un bouton Quit. Pour lancer ensuite le Finder depuis Wings, il suffit de lui faire un bouton ad-hoc dans ses écrans. Ce patch comprend plusieurs temps:

- Remplacer la chaîne hexa *[AD FA 7E 8D 65 86 22 A8 00 E1 29 20]* par *[AD FA 7E 9C 65 86 22 A8 00 E1 29 20]*
- Remplacer (quelques octets plus loin) la chaîne en Ascii bas *[*:System:Finder]* par *[9:Wings.Quitter]*
- Remplacer la chaîne *[Finder]* du bouton de Wings par la chaîne *[Quit]*.
- Copier dans le même sous-catalogue les fichiers *Start* et *Wings.Data* du disque Wings, et y joindre le fichier *"Wings.Quitter"* de la disquette de ce numéro. Renommer *Start* en *Wings*, et Wings est devenu une application normale, comme Prosel, le Finder, le lanceur de TransProg, ou tous les autres lanceurs. Enfin, *presque* normale: elle se quitte simplement par le bouton Quit au lieu de se quitter par ⌘-Q.

Si ce ⌘-Q qui reboote au lieu de quitter vous gène, remplacez-le par une autre combinaison de touches que vous ne risquez pas de taper par inadvertance: je suppose que vous savez faire ce type de patch, sinon faîtes-le nous savoir.

Le fichier *"Wings.Quitter"* se résume à un simple appel Quit P16: je ne vous mets pas le source!

Ce que ça fait.

Tout le patch se joue sur le fait que la différence entre un Quit avec retour ensuite et un Quit sans retour ensuite sont deux commandes GS/OS identiques, sauf sur un bit d'un paramètre. Les passages patchés se liraient ainsi:

LDA Variable
STA ParamètreQuit
JSL GS/OS ($E100A8)
    $2029 ; Quit GS/OS
    $XXXX ; pointeur des paramètres de Quit

Il suffit donc de remplacer le *STA ParamètreQuit* par un *STZ ParamètreQuit*, et le tour est joué. Dans le second patch, on change aussi le fichier désigné (fichier appelé Finder dans le catalogue /System du disque de boot, préfixe *) par le nom d'un fichier "Wings.Quitter" situé dans le même catalogue que l'application (préfixe 9 de GS/OS).

Utilisation de l'outil de SynthLab :

MidiSynth Player

Jean-Pierre Charpentier

Maintenant que l'ensemble est officiellement vendu par l'APDA, on s'en rend mieux compte: le programme **SynthLab**, que nous admirons tant, est une démonstration de toutes les possibilités sonores qui nous sont offertes par ce qui est le vrai programme, **MidiSynth, le Tool 35**. Avec cet Outil 35, il est maintenant possible d'introduire dans nos programmes, avec facilité, des musiques de qualité tirant toute la quintessence de l'Ensoniq (ou DOC, la puce-son du GS).

J'ai écrit le programme **MidiSynth.Play** comme un exemple d'utilisation de ce Tool 35. *MidiSynthPlay* permet simplement de charger et d'écouter des musiques tout en faisant varier certains paramètres.

Le Tool 35 est composé de trois parties: Synthétiseur, Séquenceur, Midi. Cette dernière fonction est volontairement laissée de côté pour cet exemple. En conséquence, vous n'avez pas besoin du CDEV Midi.

Pour les utilisateurs

Pour faire fonctionner **MidiSynth.Play**, il faut bien entendu que vous ayez *l'ensemble logiciel MidiSynth:* il est en vente dans le commerce. C'est une dépense, mais franchement, **si vous n'avez pas cet outil, vous ne savez pas ce que signifie le "S" de "GS".**

Une fois que vous avez cet ensemble, il faudra commencer par mettre l'outil MidiSynth (*TOOL035*) dans le sous-catalogue */SYSTEM/TOOLS* de votre disque système, exactement comme vous l'avez déjà fait pour les deux premier Tools sonores de ToolBox Mag: sans quoi *MidiSynth.Play* refusera de démarrer. Il faut aussi (est-il besoin de le rappeler?) que vous tourniez **sous 5.02 minimum**.

Pour écouter les musiques SynthLab, il faudra également vous faire un disque sur lequel vous aurez copié dans le même catalogue tous les fichiers *"XXX.Seq"*, *"XXX.Bnk"* et *"XXX.Wav"* des disquettes MidiSynth. *MidiSynth.Play*, dans sa version présente, attend que ces trois types de fichiers complémentaires soient dans le même dossier.

Il ne vous restera plus alors qu'à lancer l'application *MidiSynth.Play* du sous-catalogue */Babar* de la disquette ToolBox Mag 4. Tout en écoutant les musiques SynthLab, vous pourrez varier le tempo de la musique et changer le rythme du métronome intégré. Vous pouvez à tout instant arrêter la musique avec le bouton "[]", et la reprendre avec le bouton ">". Je vous laisse trouver tout seuls ce que fait l'option "changer bordure" et les boutons qui l'accompagnent.

Pour le programmeur

Les ressources du programme, et même le squelette de départ de l'application, ont été réalisés avec *Genesys 1.2* (fichier *DF.REZ*). *Attention aux fantaisies de Genesys avec la ressource Close (MenuItem 107) si vous rajoutez un item au menu.* Pour réassembler (assembleur APW/Orca), il suffit de lancer *DF.MAKE*.

Le fichier source *DF.SRC* faisant 64k (**64k juste**, l'éditeur d'Orca/APW vous le dira), vous ne pourrez introduire de modifications qu'en le tronçonnant. Pour la même raison, ToolBox Mag sera sans doute incapable de le passer dans les colonnes de la revue. Vous le trouverez donc dans le sous-catalogue */BABAR* de la disquette: je ne mets ici qu'un sommaire, un synoptique du programme.

Trois points à signaler

• Pour arrêter la musique temporairement, par exemple lorsqu'on va lire une disquette, utiliser les appels *_MSSuspend* et *_MSResume*.

• Les notions de Beat et de Mesure sont entièrement virtuelles (comme les barres de mesure dans MusicStudio).

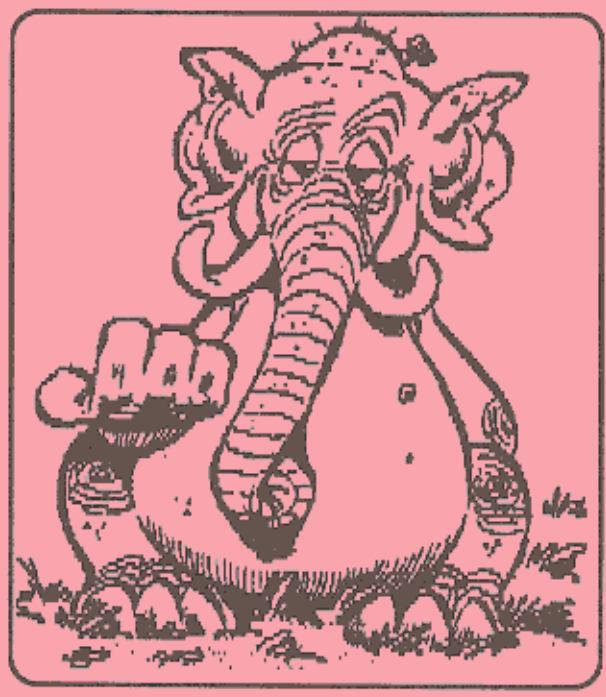
• Ne pas utiliser *_GetMSData* et *_SetCallBack* pour une lecture simple. Ca ne sert à rien et consomme du temps processeur.

Synoptique du programme MidiSynth.Play :

Vérification de la présence du Tool35. Démarrage du Tool35 par *_MSStartUp*

Sélection de la séquence par l'utilisateur "MyFile.Seq".
Chargement de "MyFile.Seq".
Dans "MyFile.Seq", récupération de la chaîne "MyFile.Bnk".
Chargement de "MyFile.Bnk".
Dans "MyFile.Bnk", récupération de la chaîne "MyFile.Wav".
Chargement de "MyFile.Wav".
Dans "MyFile.Seq", récupération du Tempo.
Fixation par *_SetTempo*
Fixation des Tracks à jouer par *_SetPlayTrack*.
Fixation de la relation Track/Channel, donc Track/Instrument, par *_TrackToChan*.
Redirection des Tracks par *_SetTrackOut* (vers le synthétiseur uniquement).
Dans "MyFile.Bnk", récupération des Instruments Records.
Fixation par *_SetInstrument*.
Dans "MyFile.Wav", récupération des PCM Data.
Recopie des PCM Data dans la DOC-Ram par *_WriteRamBlock*.
Fixation de l'Item de démarrage dans la séquence par *_Locate*.
Démarrage de la musique par *_SetPlayer*.

Fermeture du Tool35 par *_MSShutDown*.



Premier portrait officiel de Jean-Pierre Charpentier Après la soutenance de sa thèse

Note aux programmeurs en Genesys

Deux points à signaler aux utilisateurs du magnifique Genesys 1.2:

• Il est parfaitement possible d'éditer **avec Genesys lui-même** les ressources qu'utilise par défaut Genesys dans l'option "New Application" (fichier */GENESYS/GEN.WORK/DEFAULTFILE*). Si ce n'est pas déjà fait, veuillez renommer votre *DEFAULTFILE* en *DEFAULTFILE1*, et recopier le *DEFAULTFILE* du catalogue */DIVERS/* de la disquette ToolBox Mag 4 en */GENESYS/GEN.WORK/*: ainsi vos applications futures diront automatiquement "Fichier" et non "File", "A propos" et non "About", etc.

• Rien à faire, Genesys se bloque la touche Option pour ses commandes Option à lui, et force le clavier en américain. C'est discutable, sachant qu'un des usages principaux de Genesys est précisément pour la traduction des ressources.

En attendant une version de Genesys où ce blocage du clavier sera une option, il est quand même possible (et souhaitable!) de mettre nos caractères français dans les ressources éditables par TextEdit (avec LineEdit, ça ne marche pas). Simplement, nous voilà ravalés au rang de vulgaires utilisateurs de Macintosh: il faut faire du copier-coller depuis un accessoire. **Cet accessoire existe**, c'est l'excellent **Stricto Sensu GS d'Yvan Kœnig** publié par le regretté Pom's.

JYB.

Le plus grand réseau du monde

Patrick Pointu

[NDLR. Dans un monde informatique qui porte de plus en plus l'uniforme bleu, il existe des gens qui veulent maintenir le rôle de l'ordinateur comme instrument de la liberté des individus: ils ont souvent un Apple II GS.
Dans un monde où les communications sont de plus contrôlées par d'énormes bureaucraties, entreprises ou armées, il existe des gens qui exercent encore leur droit de communiquer librement et gratuitement entre individus: ce sont les radioamateurs.
ToolBox Mag est donc un lieu naturel d'expression de ceux qui font à la fois l'un et l'autre, et serait heureux de susciter quelques vocations. Vous trouverez sur la disquette ToolBox Mag 4, dans le sous-catalogue :RADIO.AMATEURS, un fichier d'introduction générale à la pratique du radio-amateurisme (OM.INTRODUCTION), et deux fichiers d'adresses (GS.AMAT et OM.ADRESSES) de référence. Tous ces fichiers (fichiers textes GS) sont de Patrick Pointu.]

"Your report is five nine, five nine; the QTH is Paris, Paris; how are you receiving me? Foxtrott Oscar five X-ray Zoulou, this is Foxtrott Delta one Oscar Romeo Foxtrott; over."
Voilà ce que vous entendriez si vous écoutiez les ondes courtes où des millions de radioamateurs se retrouvent uniquement pour le plaisir de se dire bonjour. Dans l'exemple ci-dessus, un "OM" parisien explique à son homologue tahitien qu'il le reçoit très bien (1er chiffre [1 à 5]: compréhensibilité du message, 2ème chiffre [0 à 9]: force du signal).
Mais la phonie n'est que la facette la plus connue de cette activité. Saviez-vous que le morse est encore utilisé par des milliers de fanatiques, que la Lune est utilisée comme un miroir pour renvoyer les ondes de l'autre côté du globe, que des satellites radioamateurs tournent actuellement au dessus de nos têtes et surtout que **certains de ces passionnés utilisent un Apple IIGS?**

Dans toutes les possibilités offertes au radioamateur, l'ordinateur peut prendre une place non négligeable.
Ainsi, il peut se charger de la gestion du "log": journal de bord obligatoire dans lequel on marque la date, l'heure, la fréquence et le mode utilisé pour chaque contact. Le programme se chargeant alors de retrouver si l'on a déjà établi le "QSO" (liaison radio) précédemment et d'imprimer les "cartes QSL" qui sont envoyées aux correspondants pour immortaliser chacune de ces transmissions. On pourra ensuite, toujours grâce à l'ordinateur, rechercher le nombre de pays dont on a reçu la carte ou bien rédiger un compte-rendu de concours en calculant automatiquement les distances entre chaque station.

En-dehors du GS, de quel matériel auriez-vous besoin pour faire de la radio amateur?
Tout d'abord, l'organe principal est l'**émetteur-récepteur**, éventuellement un récepteur si vous n'envisagez de faire que de l'écoute. Un poste grandes ondes ne convient pas, il faut absolument un récepteur de trafic décamétrique, si possible à couverture générale de 1 à 30 MHz pour recevoir aussi les agences de presse.
Il vous faut ensuite **un modem**. Il en existe autant de types que de modes de transmission: RTTY, Packet-Radio, Fax, CW, Amtor... Vous pouvez aussi trouver des modems multimodes. Ils se branchent généralement sur une prise prévue à cet effet sur l'émetteur-récepteur et sur une prise RS-232.
Le radioamateurisme étant un loisir expérimental, vous trouverez facilement des schémas de décodeur monomode ou même des kits complets pour quelques centaines de francs, certains étant sous forme de cartes à insérer dans un slot de votre Apple.
Enfin, du côté de l'Apple IIGS, vous aurez besoin d'**un logiciel de communication**. On en trouve une dizaine en freeware ou shareware (Apr, Tic.Com, freeterm...).

Une fois équipé, il vous faudra choisir votre mode de transmission. Il en existe des dizaines et des dizaines, qu'on peut classer en trois catégories:
• **La transmission de la parole**: Les modes couramment utilisés sont la Modulation de Fréquence (FM), la Modulation d'Amplitude (AM) et la Bande latérale unique (BLU ou SSB en anglais).
• **La transmission de textes écrits**: Les modes sont nombreux: La CW (le morse!), le RTTY, le Packet Radio, Baudot... Le packet radio est le mode le plus en vogue actuellement dans ce type de transmission. Il s'agit d'un dérivé de Transpac adapté à la radio. Il s'est mis en place un réseau de BBS (Bulletin Board Systems) qui permet de laisser un message à n'importe quel OM du monde entier grâce à l'interconnexion de ces boîtes aux lettres. On peut aussi télécharger des programmes, lire des informations sur la propagation, laisser des petites annonces pour du matériel électronique, demander de l'aide au sujet de tel ou tel programme...
C'est le plus grand réseau du monde, et c'est gratuit!
Le seul investissement consiste à acheter un modem packet ou bien à le construire soi-même. Pour ce qui est du RTTY, vous pourrez décoder les agences de presse du monde entier, pourvu encore une fois que vous ayez l'émetteur-récepteur et un modem ad-hoc.
• **La transmission d'images**: Pour les images fixes vous avez la SSTV et le Fax. Pour les images animées, vous avez la TVA (Télévision Amateur bien sûr!). Là, l'Apple peut vous aider moyennant un interface plus complet.

Aujourd'hui, la plupart des radioamateurs informatisés sont des galériens. De notre côté, la plupart de nos programmes datent de l'Apple II+! Les quelques programmes en mode natif ne font pas le poids devant la multitude de logiciels pour brouettes. Mais bon sang, GS ca veut dire graphisme et son. **La radio c'est aussi du graphisme et du son.** Vous ne trouvez pas qu'il manque quelquechose entre les deux? Voilà pourquoi, quelques GSmaniaques ont décidé de se regrouper au sein du GSAMAT pour essayer de faire avancer tout ça.

Si vous avez envie d'entrer en contact avec le plus grand réseau du monde, contactez-nous:
GSAMAT - c/o M. Patrick POINTU - 19, av Léopold II - 75016 Paris.

Made in France

Bernard Fournier

Breakers

Oui mon frère, voici venir
le RAP sur ton GS d'avenir.
C'est super, et ça décoiffe,
car Breakers est super classe !

Un jeu de fort belle facture, et bien dans l'air du temps. Deux personnages sont sur l'écran: le premier effectue des mouvements de RAP et vous devez effectuer les mêmes avec le deuxième personnage (à l'aide des touches 7-8-9-5). En fond sonore, votre outil 219 (voyez la dernière version sur la disquette de ce numéro) assurera le rythme! Le jeu est de difficulté croissante (de plus en plus de mouvements à répéter) et renouvelle donc fort intelligemment le style SIMON. Outre ses qualités pédagogiques de mémorisation et de coordination motrice, ce jeu est un bel exemple de programmation en Pascal et est disponible gratuitement.

Francis Wolkowitch,
4 bis rue de Lyon, 75012, Paris

91 : l'année Liautard ?

Tout d'abord, le mois de Janvier vit se dérouler à Cormeilles une réunion des passionnés du GS, à l'initiative de notre spécialiste ès carte Overlay. Placée sous le signe de l'œcuménisme, cette réunion fut fructueuse à bien des égards: en premier lieu beaucoup de monde vit encore plus de monde, ce qui est en soi réjouissant. Et puis ce fut l'occasion de compter les nouveaux MacIntoshiens: à en juger par le seul Mac présent et la frileuse poignée de ses admirateurs, le GS garde encore pour longtemps ses fidèles !
Que ce passa-t-il de notable à cette réunion en ce qui concerne les logiciels ? On y vit quelques exclusivités: Appleworks GS en cours de francisation (dictionnaire compris), HyperStudio également en cours de francisation (ces deux produits devant en principe être disponibles sous peu). On y vit également les spectaculaires productions du FTA (Nucleus, Modulæ et XMas Demo) ainsi que la prolifique production de l'organisateur de ce week-end.
Il nous présenta (outre ses charmants bambins) ses dernières œuvres - gratuites - disponibles sur simple demande à l'adresse suivante:

Jacques Liautard,
13 avenue de la Libération,
95240 Cormeilles en Parisis

• **Horloge**: cet été il sera du dernier chic de se promener à la plage avec son GS au poignet: cet utilitaire affiche en effet une magnifique horloge parlante. Succès garantis sur les plages ! et au diable la montre Helvète !

• **Moulisons 1.2**: cet autre utilitaire permet la conversion des sons Hyperstudio en ressources pour Hypercard GS. Il permet également le compactage de ces sons et la modification de certains paramètres (volume, fréquence). Sont également reconnus les sons au format AE et RawData.
Le menu offre une possibilité de compactage des images: cette option n'est pas opérationnelle... Enfin, bon, vous êtes au courant; alors ne perdez pas deux heures à chercher comment compacter une image avec cet utilitaire !

• **Point-Point**: avec cet utilitaire on doit relier une suite de points de manière à former un dessin. Ce jeu est très connu des lecteurs de magazines d'enfants: on relie les points 1 à 32 et on voit apparaître une figurine connue. Là c'est la

Rectificatif

Dans le n° 3 de ToolBox Mag, je signalais dans cette même rubrique que François Uhrich proposait TransProg **et** Fonte-DA Installateur pour le même prix... Avant que François n'exige ma comparution devant un tribunal populaire, je tiens à lui présenter mes excuses et par la même occasion je vous demande la plus grande indulgence: **TransProg et Fonte-DA Installateur sont deux sharewares totalement distincts,** le premier pour 90 F et le second pour 60 F. Et ce sont **tous les deux des indispensables** pour tout utilisateur de GS.

BF

même chose, mais sur le GS ! et bien entendu on peut créer soi-même ses formes à relier. Un plus apporté par l'informatique: les points à relier sont en fait le contour d'une image qui apparait une fois la figurine complétée. Des options permettent d'aider l'enfant dans son tracé et rendent ce logiciel utilisable même par les tout-petits petits.

Un grand bravo à Mr Liautard qui prépare ainsi des hordes de futurs passionné(e)s de l'informatique !

Fonction Pro 1.02

On annonce immédiatement la couleur: c'est en Basic et en DHGR.
Bien, vous êtes encore là ? Alors détaillons ce logiciel. Il permet d'entrer une fonction mathématique et de tracer automatiquement sa représentation graphique sur un intervalle donné. On peut en outre modifier tous les paramètres et le facteur d'échelle. Bref un outil bien pratique pour les 'matheux' et qui manquait cruellement sur notre GS.
Un reproche cependant: l'interface utilisateur est assez médiocre et la documentation un peu légère ne contribue pas à simplifier les choses. Ceci étant, cet outil est très performant et rendra sûrement d'immenses services à pas mal de potaches.
Ce logiciel est également gratuit et est disponible auprès de:

Mr Rondof Jean-Paul,
11 rue du Frais Foyer,
49000 Angers.

Un jeu en six versions :

Fun Beyond GS

Olivier Goguel

Bon, ce ne sont quand même pas ces satanées consoles qui vont nous empêcher de nous continuer à nous amuser avec notre GS! Puisque la production de jeux par les "Big Companies" est un peu en baisse en ce moment, le FTA et ToolBox Mag prennent la relève. Ayant vu passer furtivement l'idée sur une brouette (il semble que l'idée de départ vienne d'un jeu sous Unix de J.Geertson appelé "xcolumns"), nous avons décidé de l'implanter sur une machine pour les grands.

Le principe du jeu est une variante de Tetris: au lieu d'entasser des blocs de façon à boucher les trous, on prend des figures composées de trois blocs carrés de couleur. Ces blocs, une fois tombés, s'effacent dès que trois d'entre eux de la même couleur sont alignés, selon les règles du morpion (alignement horizontal, vertical, diagonal dans les deux sens). Du coup, les blocs qui sont au-dessus tombent dans les trous ainsi dégagés, par "gravité": éventuellement, ils disparaîtront alors à leur tour, par réaction en chaîne.

On ne peut pas, comme dans Tetris, faire pivoter la figure, mais on peut inverser l'ordre des blocs (touche R). La touche N permet d'afficher la pièce suivante. Un "Help" intégré vous dira tout ce que avez à savoir pour jouer à Fun Beyond GS.

Originalité, Fun Beyond GS est proposé en versions anglaise et française, et sous trois formes à chaque fois: application Prodos 8, application GS/OS, CDA. Vous choisissez votre version à l'assemblage, qui se fait directement en une seule fois. Evidemment, c'est avec Merlin 16 qu'on fait tout ça! Vous trouverez le source complet sur la disquette, catalogue /BEYOND, sous-catalogue /BEYOND.SOURCE.

GS/OS, Tais toi ! Un extrait du source de

Fun Beyond GS, par Olivier Goguel

[NDLR: Nous n'avons pas la place dans la revue pour imprimer le source entier d'Olivier. Vous le trouverez donc dans le catalogue /BEYOND de la disquette ToolBox Mag 4. Vous pourrez ainsi rajouter à Beyond GS l'acceptation des minuscules du clavier, par exemple. Mais nous tenons à imprimer une routine, marginale par rapport au jeu, mais particulièrement intéressante: Olivier s'y est battu avec GS/OS pour lui faire faire ce qu'il voulait lui, et non ce qu'Apple avait prévu. Et devinez qui a gagné ?
Il nous semble en revanche qu'Olivier s'est inutilement battu avec P8 pour le chemin d'accès de son programme: à notre connaissance, le chemin d'accès complet d'une application SYS qui vient de se lancer est toujours en 00/0280 et la suite. Quelqu'un sait-il si c'est juste une habitude, ou si cela est garanti par le Bureau Politique?]

Dans tout ce programme, nous allons -comme d'habitude- tout gérer par nous mêmes (affichage, gestion des erreurs, etc). Malheureusement, nous allons être légèrement gênés par GS/OS, qui a la possibilité d'afficher des fenêtres de dialogue (texte ou graphiques) pour nous avertir de l'apparition d'erreurs éventuelles. Théoriquement, l'appel SetSysPref permet de dire à GS/OS de ne pas s'occuper de la gestion des erreurs:

```
JSL SE100A8
DA $200F          GetSysPref
ADRL SysDefaultParms

JSL SE100A8
DA $200C          SetSysPref
ADRL SysParms
```

Mais en fait, dans certains cas, GS/OS va -malgré les préférences demandées- afficher certaines fenêtres d'erreurs: ce qui serait fatal à notre application aussi bien qu'au système, dont la cohabitation est impossible hors des limites définies par... votre serviteur !
Nous sommes alors obligés de patcher GS/OS, sans scrupule, puisqu'aucun appel homologué ne permet de faire exactement ce que nous voulons. On ne va tout de même pas se laisser dominer par un système d'exploitation quelconque...
J'ai donc "examiné" une partie de GS/OS pour remonter jusqu'à la routine qui m'intéresse: celle-ci se trouve pointée par un vecteur (dont l'adresse, quoique non déclarée, me paraît inamovible) en $01FC98.
Si on suit ce qui est pointé par ce vecteur, on trouve un bout de routine qui commence par :

```
*        LSR
*        BCS  AfficherFenetre
*        LDAL $00B9F6        Test des préférences
*        BMI  AfficherFenetre (Bit15=Affichage)
*        ...
*        RTL
* Aff_Fenetre ...
```

On voit bien le BCS qui peut forcer l'affichage d'une fenêtre malgré l'état du drapeau Préférences...
Nous allons donc détourner le vecteur en $01FC98 vers une routine qui remet l'accumulateur à 00, pour que le BCS ne soit jamais pris, et ainsi on n'aura jamais de fenêtres indésirables à l'écran...
Nous enlèverons notre patch au moment de quitter, bien entendu. En attendant, c'est notre application qui gérera les erreurs,en l'absence de tout message GS/OS (en particulier du message "Insérez le disque XXX").

```
         SEP $20
         LDAL SE1C068
         PHA
         LDAL SE1C08B
         LDAL SE1C08B

         REP $30

         LDAL $01FC98        On vérifie que c'est
         AND #$00FF          bien un vecteur...
         CMP #$5C
         BNE NoPatch

         LDAL $01FC99        Sauvegarde de
         STA SaveVect+1      la valeur du vecteur
         LDAL $01FC9A
         STA SaveVect+2

         LDA NewVect+1       et détournement
         STAL $01FC99        vers notre routine
         LDA NewVect+2
         STAL $01FC9A

NoPatch  SEP $20
         PLA
         STAL $E1C068
         REP $30

         BRA Continue_Init
```

```
SysDefaultParms DA 1        Sauvegarde
         DA 0               des préférences
SysParms DA 1
         DA 0               GS/OS, tais-toi !

NewVect  JML NewFC98        nouvelle valeur FC98
NewFC98  LDA £0
SaveVect JML $000000        Ancienne valeur FC98
```

(La suite sur la disquette...)

GS News

Eric Weyland

Ce qu'il faut savoir

HyperCard GS (Apple Computer) : compléments

C'est pendant la dernière AppleFest qu'Apple a officiellement présenté la version couleur d'HyperCard. Ralph Russo lui même a fait une brillante démonstration du puissant logiciel hypermédia.

HyperCard GS est un produit Apple et non pas un produit Claris. Il est disponible depuis le 15 Février auprès de tous les revendeurs Apple. L'ensemble est composé de six disquettes et trois manuels. Une des disquettes contient le programme proprement dit et la pile HOME. Une autre contient une pile d'aide ; l'aide dans HyperCard est très bien faite : elle est très détaillée et explicite dans le fonctionnement du programme. Les autres disquettes contiennent des fichiers de commandes externes (XCMDS), des fichiers de fonctions externes (XFCNS), un éditeur d'icônes, un éditeur de sons.

Le produit final est livré avec trois manuels de référence :

1) Débuter avec HyperCard GS,
2) HyperCard GS Référence, et
3) Le manuel HyperTalk (le langage de programmation d'HyperCard).

Aux USA, HyperCard GS n'est pas livré avec les configurations Apple II GS neuves, car le programme a absolument besoin de 1,5 Méga de mémoire et d'un disque dur pour tourner. HyperCard GS est livré avec le système d'exploitation GS/OS version 5.04.

C'est en fait un HyperCard Mac version 1.2.5, mais avec des enrichissements considérables (la couleur, mais aussi la gestion de data en base de données, etc).

HyperMover (une pile Macintosh et une pile Apple II GS, encore en version bèta pour le moment) sera distribué ultérieurement en version définitive par l'APDA. Il permet de décompiler des piles HyperCard Macintosh en les transformant en fichiers intermédiaires. On peut alors transférer ces fichiers intermédiaires sur Apple II GS pour reconstituer une pile HyperCard GS. Ce procédé n'est pas très rapide, et il faut disposer à la fois d'un Macintosh et d'un Apple II GS, car HyperCard GS ne peut pas lire directement les piles venant du Mac. Si la pile Macintosh fait appel à des commandes ou à des fonctions externes, celles-ci devront être ré-écrites ou recompilées sur Apple II GS.

HyperMover n'est en fait utile que pour les piles Mac contenant beaucoup de programmation HyperTalk, car de toute façon l'interface utilisateur est à refaire, ne serait-ce que pour la couleur. Ce qui se profile en fait, et qui est beaucoup plus intéressant, est le développement simultané, dès la conception, de piles à la fois pour Mac et pour GS.

Publicité : Apple a les moyens

Quand Apple veut faire savoir quelque chose cela ne passe pas inaperçu. C'est plus de 40 millions de dollars qui ont été investis pour promouvoir le nouvelle gamme de Macintosh. Cela a porté ses fruits car il y a déjà des ruptures de stocks sur ces modèles ; les revendeurs Apple en manque de Macs pourraient en profiter pour essayer de vendre des Apple II GS plutôt que des gamelles compatibles IBM...

GS/OS 5.04 (Apple Computer)

Le système 5.03 ayant été retiré de la vente car le driver d'ImageWriter avait des faiblesses sur des configurations pauvres en mémoire, c'est le système 5.04 qui est le dernier système en date aux Etats-Unis. (pour la France, voir page 4). Pour avoir la version US, il suffit d'acheter HyperCard ou Genesys 1.2.

Poisson 8 bits

Voici quelques croustillantes particularités de la carte d'émulation Apple IIe destinée par Apple à boucher l'unique slot du tout nouveau Macintosh à bas prix. Cette carte ne peut

pas reconnaître plus de 1 Méga de ram (le LC de base a 2 Mégas en ligne) ; la raison invoquée est que les cartes d'extension *Apple* pour Apple IIe n'ont jamais dépassé cette limite! Cette carte peut être configurée en 1 ou 2 mhz ; le programme fourni avec la carte fera une pseudo gestion de slot par simulation logicielle. Il est possible d'y connecter un joystick, un lecteur 5,25 et bien sûr un lecteur 3,5. Disponibilité **Avril 1991: c'était donc un poisson !**

Broderbund Software : Pas dans le coup !

Cette très réputée société d'édition américaine vient de décider de ne pas sortir de version Apple II GS de Prince of Persia qui est le must du jeu d'arcade-aventure sur les Apple 8 bits. Influence rétro d'Apple?

Harmonie : L'ImageWriter fait bonne impression !

La série de drivers d'imprimantes éditée par la firme Vitesse contient un driver pour ImageWriter II tout à fait performant : c'est une nouvelle version de celui qu'évoquait Jean-Yves Bourdin dans le numéro 1 de ToolBox Mag. Ce driver a été écrit par Bill Heineman.

Dragon Wars : Pas anonyme...

L'auteur de Dragon Wars (voir numéro précédent) n'est pas un inconnu : il s'agit de Bill Heineman, l'auteur de différents drivers. Cet homme est très occupé : il s'occupe de très près du magazine GS+, il vient de terminer un petit jeu pour ce même magazine (*Night of the living Macs* : le but du jeu est de tirer sur des Macintosh afin qu'ils tombent dans l'oubli...), il travaille pour Vitesse (Harmonie), et est employé à temps complet par InterPlay (l'éditeur de Dragon Wars). Bill projette d'écrire deux nouveaux jeux pour Apple II GS *FutureWars GS* et *James Bond Stealth Affair*; InterPlay attend de voir ce que va rapporter *Dragon Wars GS* avant de donner son feu vert...

Genesys : y a de la ressource !

Le plus performant des éditeurs de ressources est maintenant livré dans sa version 1.2, sous GS/OS 5.04. Vous en entendrez souvent parler dans ToolBox Mag: c'est vraiment un achat à envisager.

Roger Wagner : chômage ?

L'éditeur d'HyperStudio, le premier système hypermédia sur Apple II GS, n'est pas trop inquiet de la concurrence que va lui faire HyperCard. Il fignole son produit porteur en le dotant de diverses extensions : des commandes externes pour exécuter à partir de HyperStudio des programmes écrits dans divers langages (il existe pour le moment 18 XCMDS pour HyperStudio dont une qui permet de jouer des musiques de type SoundSmith), des clip-Art, et des clip-Sounds. Durant l'AppleFest, HyperStudio a fait grande impression. En effet, le programme contrôlait l'action d'un jeu vidéo intéractif : *Dragon's Lair*. Le tout tournait sur Apple II GS, muni d'un lecteur de disques optiques et d'une Overlay Card. A l'avenir, les fanatiques du joystick vont devoir investir pour pouvoir se défouler sur leur Apple II GS.

Trophées AppleFest : les meilleurs produits américains du moment

Un jury s'est réuni durant la dernière AppleFest US pour attribuer des trophées. Ce jury était constitué de représentants, des magazines : A+/Incider, A2-Central, GS+ Magazine, Nibble, d'Apple Computer, d'America On-Line, de Compuserve, et de GEnie. Plusieurs catégories ont été prévues pour l'attribution des trophées. Voici les résultats :

Meilleur programme éducatif: **Katie's Farm**
Meilleur programme Apple II (8 bits) : **Proterm 2.2**
Meilleur programme Apple II GS (16 bits): **HyperStudio 2.1**
Meilleur programme Freeware ou Shareware : **ShrinkIt GS**
Meilleure innovation : **Ramfast SCSI**
Meilleure réalisation multimédia : **HyperStudio 2.1**
Meilleur outil de développement ou langage de programmation : **Genesys 1.2**
Meilleur utilitaire de débugging : **GSBug 1.5**
Meilleur périodique sur l'Apple II : **A2-Central**
Meilleur serveur : **America On-Line**
Hardware de l'Année : **Carte SCSI High Speed DMA**
Programme de l'année : **HyperStudio 2.1**

Une mention spéciale du jury a été décernée à **Appleworks 3.0**. Il serait temps de faire une AppleFest mondiale, histoire de faire concourir les produits français...

Suprême (Vitesse)

C'est sous le nom de Suprême que l'ensemble des programmes de chez Vitesse sont regroupés pour être vendus en lot. Il s'agit de : Salvation, Renaissance, Exorciser, Wings et Deliverance.

Au fait, savez-vous pourquoi Vitesse tient à donner ainsi des noms français à ses produits? Eh bien, d'une part, c'est du dernier snobisme aux USA en ce moment. D'autre part, un

mot anglais sur deux aux Usa est une marque déposée. On va donc piocher chez nous: bref, ils font comme ToolBox, mais dans l'autre sens!

Le Hard

Zip Chip GS : mauvaise nouvelle

Cette carte accélératrice, n'est dans son état actuel, toujours pas totalement compatible avec AppleTalk. Fâcheux... Une bonne nouvelle quand même : un co-processeur arithmétique serait en cours de développement pour cette carte. D'autre part, Zip Technologies fait de bonnes affaires : tout leur stock de Zip Chip GS a été vendu durant l'AppleFest.

Lecteurs Æ 3,5 et Æ High density (Applied Engineering)

Selon Phil Montoya d'Applied Engineering, deux nouveaux lecteurs de disquettes sont maintenant disponibles. Il s'agit d'un lecteur 800k (Æ 3,5) destiné à concurrencer le lecteur Apple, et d'un lecteur haute densité 1600k (Æ High density). Il est possible de transformer un lecteur Æ 3,5 en lecteur Æ High density par un simple remplacement de circuit intégré (et par l'adjonction d'un autocollant HD sur la face avant du lecteur...).

L'avantage du lecteur haute densité, outre sa double capacité, est son aptitude à pouvoir accepter les disquettes MS/DOS haute densité quand on le branche sur une PC Transporter. Attention ! Pour utiliser ce lecteur haute densité sur un Apple II GS, il est nécessaire d'utiliser un driver, à placer sur votre volume de boot. Ce driver est fourni avec le lecteur, et prend la place du driver Apple, fourni avec le GS/OS, chargé de la gestion du lecteur Apple 3,5.

Ce nouveau driver est capable de gérer les lecteurs Apple, de façon plus rapide : ainsi des configurations ayant ces deux types de lecteurs sont parfaitement envisageables. Ce nouveau driver fonctionne sous ProDos 8, mais il faut charger GS/OS au préalable. Un boot direct sur ProDos 8 ne fonctionne pas avec le lecteur Æ High density.

Voici les formats reconnus par ses nouveaux lecteurs : au format ProDos exlusivement - 800 Ko 2:1 - 800 Ko 4:1 - 400 Ko 4:1 - 1600 Ko 2:1 - 1600 Ko 4:1. Des FST (File System Translator) pour Macintosh ou MS/DOS-beurk ne sont pas disponibles, et ne sont pas prévus.

Selon Æ, cela est dû aux limitations du circuit de l'Apple II GS contrôlant les lecteurs de disquettes (IWM : Integrated Wozniak Machine). Le lecteur haute densité ne fonctionne que sur Apple II GS ; il ne fonctionne pas sur les Apple 8 bits.

D'ores et déjà, Prosel 16 reconnaît ce lecteur pour faire ses back-ups.

Avec ce lecteur Æ, et la baisse du prix du lecteur Apple similaire, l'argumentation d'Yvan Kœnig dans ToolBox Mag 2 ne tient plus tout à fait. Reste quand même que, si vous avez besoin d'un lecteur de disquettes de plus grande capacité pour le GS, vous avez intérêt à comparer avec un Syquest, et avec les lecteurs 3,5 haute densité SCSI dont nous vous parlerons dans un prochain numéro. Rappelons qu'une disquette 3,5 1 600k Haute Densité est la même qu'une disquette 800k, mais avec un deuxième trou, à droite. Vous avez donc compris comment vous faire des disquettes 1 600k. Attention, cependant, ces trous doivent être carrés pour être reconnus par les lecteurs Haute Densité. Une simple perceuse à mèche carrée suffit. Contactez Hubert Loiseleux à ToolBox si vous n'en trouvez pas chez votre quincailler.

SendFax (Applied Engineering) : un fax dans votre Apple II GS

Là aussi, il s'agit d'un matériel qui fonctionne exclusivement sur Apple II GS. Il s'agit d'une série de circuits électroniques venant en complément des modems Datalink Express et Datalink GS. Un programme est fourni ; il permet d'accéder aux possibilités de télécopie. C'est en fait un driver de modem que l'on peut sélectionner dans le Control Panel graphique. Quand vous êtes dans votre application, composez votre message, et faîtes une impression. Il est alors possible d'entrer un numéro de téléphone. Le message est ensuite envoyé sur la ligne téléphonique via le modem.

Attention, donc, ce n'est pas un fax complet: ça envoie, mais ça ne reçoit pas. Pour ses fax, ToolBox a... un fax, tout simplement. Dédier un ordinateur à un fax ne se justifie que pour qui fait, pendant la nuit, aux heures creuses, du *mailing par fax* (oh, si, c'est courant, hélas!).

Enhanced Vision Plus (New Concepts) : Réincarnations

C'est une carte permettant de digitaliser des images à partir d'une source vidéo. Elle portait préalablement le nom de *Visionary GS* (Virtual Realities a disparu dans la nature), et à l'origine le nom de carte AST. Cette carte renaît donc de ses cendres. En fait, c'est le programme qui accompagne la carte qui est remarquable. Il a été écrit par Scott Gentry et

Jonash Stich. Il fonctionne en mode couleur dithering. Avec ce matériel, il faut environ 7 secondes pour capturer une image couleur. Il est bien entendu possible de faire de la digitalisation en monochrome. En prime, si vous faîtes l'acquisition de ce matériel, vous aurez droit à un programme appelé Allison 3200 qui permet de faire des conversions en images 3200 couleurs.

Le soft: les jeux

Cryllan Mission (Victory Software)

Pour ceux qui ont eu la patience de terminer Cryllan Mission, Victory Software vient de sortir un deuxième scénario. Le jeu est toujours en mode desktop.

Le soft: les utilitaires

BannerMania Broderbund Software)

Nouveau programme de la société californienne tournant sur tous les Apple II. Il permet d'imprimer toute sorte de banderoles. A défaut d'imprimante sur tissu, ça ne peut pas faire les manifs, mais cela permet de grandes banderoles papier pour une vitrine de commerçant, par exemple.

Photonix II (Toolbox) : version 2.30

Cette dernière version règle différents problèmes d'éjection de disquettes ; la fonction antivirus traite le virus Apocalypse.

Desktop Screen Saver (Roger Wagner Publishing)

Voici un autre NDA produit par Roger Wagner (le premier est Desktop File Control). Celui ci permet de capturer des images ou des portions d'image se trouvant sur l'écran graphique puis de les sauvegarder. On peut fixer un délai, de façon à sauver un écran avec menus déroulés, par exemple. Tout ça ne me semble guère vraiment nouveau...

Clip Art Volume 2 Roger Wagner Publishing)

Voici une collection de 500 images à utiliser dans HyperStudio (ou ailleurs). Tous les thèmes sont présents : les animaux, les visages, les constructions, la nature, les bombes et les missiles... Aussi incluses, des images avec des boutons et des bordures.

Golf Teacher (FYI Publishing) : HyperStudio joue au golf

Voici la première pile commerciale pour HyperStudio. Cette pile permet d'apprendre à jouer au golf. Elle est constituée de cinq disquettes 3,5. La pile utilise des graphiques et des animations. L'hypermédia permet ainsi de pallier les insuffisances des démonstrations littéraires. Cette pile commerciale met en lumière un avantage certain d'HyperStudio par rapport à HyperCard : l'existence d'un runtime gratuit permettant de lire les disquettes sans avoir à acheter HyperStudio.

Chinook Utilities

La dernière version des utilitaires Chinook (absolument nécessaires pour travailler avec un disque dur SCSI) est la 1.5. Elle est capable de reconnaître les disques durs à cartouches amovibles, et les deux versions de la carte SCSI Apple (Rom C et DMA High Speed).

Le soft: freeware et shareware

Fonds de tiroirs (Beagle Bros) : les classiques gratuits !

Les programmes suivants sont désormais gratuits ; c'est le dernier catalogue Beagle Bros qui l'annonce : Alpha Plot - Beagle Bag - Beagle Basic - Dos Boss - Big U - D Code - Double Take - Extra K - Fat Cat - Flex Type - Font Mechanic - Frame Up - Pro Byter - Pronto Dos - Shape Mechanic - Silicon Salad - Tip Disk 1 - Utility City.
Ces programmes sont sur Genie accompagnés de leurs documentations.
Aux anciens de l'Apple II, ces titres vont rappeler le temps où ils bidouillaient sur leur 2+..
Les utilisateurs du poisson d'Avril Apple pour Mac à bas prix seront contents aussi: leurs logiciels sont tellement périmés qu'ils sont gratuits!

Columns (Kenrick Mock)

Le principe du jeu est le même que **Fun Beyond GS** d'Olivier Goguel dans ce numéro. Il faut aligner des blocs de couleurs en essayant de faire des lignes de trois. Une fois qu'une ligne est formée, elle disparaît.
ToolBox Mag publie ce jeu en CDA, S16 et SYS; Columns est en mode desktop, et en shareware ce qui n'est pas le cas des autres. Je me doute que vous n'êtes pas prêts de l'acheter, puisque vous avez l'autre. A mon avis, la version FTA est très supérieure. Mais enfin, si vous voulez comparer par vous-même...

A lire

GS + (Ego System) : précision importante

Cette nouvelle revue est d'Outre Atlantique : j'avais oublié de le mentionner dans le numéro précédent.
C'est bien le seul vrai confrère de ToolBox Mag, mais il ne projette pas dans l'immédiat de version française, hélas.

Un générateur d'écrans et fenêtres texte en 80 colonnes sous P8 :

GE 80

par Régis Mange

Un nouveau générateur ou éditeur d'écrans, ce n'est pas trés original, il en existe de nombreux. Oui, mais ils sont, en général, rustiques, peu conviviaux et ne permettent pas l'utilisation des caractères mousetext. Celui-ci pourrait être un peu l'aboutissement du genre grâce à sa facilité d'emploi, à la richesse de ses fonctionnalités et, surtout à sa capacité de créer des fenêtres d'écran et de les sauvegarder avec leurs dimensions.
Il s'agit d'un outil principalement destiné à ceux qui s'adonnent à la programmation en Basic Applesoft, Pascal, Assembleur etc.. Il est entièrement développé en Assembleur 65C02, ce qui lui confère une fiabilité et une vitesse incomparables.

A quoi sert GE 80 ?

A **générer des écrans texte en 80 colonnes** débarrassés des fameux 'trous d'écrans' et récupérables dans n'importe quel environnement P8, de façon instantanée, grâce à un module de chargement relogeable selon les règles de Prodos. Il peut être aussi utilisé comme **'bloc-notes'** et utilitaire de **'gestion disque'** en-dehors de toute notion de programmation. GE 80 avec, en particulier GE.TOOLS, peut être considéré comme un **éditeur de ressources**.

GE 80 comporte deux principaux segments constituant l'architecture du fichier GE.SYSTEM et qui sont décrits ci-après:

Un éditeur d'écrans et fenêtres : GE.EDIT

Le segment 'éditeur d'écran' apporte à GE 80 des **fonctionnalités proches de celles d'un traitement de texte**, mais limitées à un écran ou à une fenêtre. On peut écrire en mode normal, inverse et utiliser les caractères mousetext, ce qui permet de générer des écrans au 'look' moderne (cadres, boîtes, flèches etc.., à l'instar de *Prosel*) et de simuler des environnements du type 'desk' en mode texte, pour une exploitation de la souris. Il offre à l'utilisateur toutes sortes de commandes très pratiques telles que:

- déplacement rapide du curseur mot par mot et dans les deux sens,
- insertion de caractères ou de lignes,
- suppression de caractères, de lignes ou de portions de ligne,
- tabulateur au pas de 5 caractères,
- écriture sur écran en vidéo inverse,
- accès direct aux caractères mousetext,
- conversion majuscules / minuscules,
- conversion affichage normal / inverse,
- écriture verticale,
- copie de lignes ou segments de lignes en insertion ou écrasement,

Il fournit 4 possibilités d'environnement de travail:

- écran/cadre avec barre des commandes en bas d'écran,
- plein écran avec barre des commandes,
- plein écran sans aucune aide,
- écran/cadre, sans barre des commandes.

On peut, de plus, lorsque le cadre est affiché, faire apparaître, en haut de l'écran, une règle de tabulation facilitant le repérage lorsque la barre de commande n'est pas présente.
La barre de commande affiche les coordonnées du curseur, le mode d'affichage, auquel est associé un type de curseur, et le mode d'écriture.
Le nombre de commandes étant élevé, l'utilisateur peut à tout moment, sans altérer le travail en cours, afficher des écrans d'aide donnant la liste et la signification des commandes et le tableau de correspondance entre les caractères ascii et mousetext.
Autre fonctionnalité intéressante: GE 80 permet de conserver en RAM deux versions d'un écran de travail. Deux tampons de la mémoire ont été réservés à cet effet. Lorsqu'on sauve sur disque la version affichée, celle-ci se projette automatiquement dans l'un des tampons sans altérer la précédente version, ce qui permet de récupérer à tout moment l'une ou l'autre version par ⇧-R.
Note technique : la mémoire-tampon principale est située à l'adresse $4000 et sert à la récupération de tout écran ou fenêtre pour les opérations d'entrée/sortie (sauvegarde, affichage, impression etc...). La mémoire-tampon secondaire est à l'adresse $D400 en 'bank 2' de la 'bank switched memory' (carte langage) en mémoire auxiliaire.
Il est également offert la possibilité de dupliquer l'écran affiché à l'aide de Option-W et de pouvoir ainsi le modifier sans perdre la version précédente en cas de remords. La récupération s'opère par un processus de bascule à l'aide de la commande ⇧-R.
De façon générale, la majorité des commandes est accessible par des séquences ⇧ ou Option.

GE.SYSTEM comporte également des **fonctions d'impression d'écran**:

- ⇧-P ou Option-P imprime l'écran sauvegardé dans la mémoire-tampon principale
- ⇧-H ou Option-H réalise un 'hard copy' de l'affichage. Cette fonction est disponible dans la plupart des niveaux de menus.

Ces fonctions ne sont effectives qu'avec une imprimante ImageWriter II en raison de l'utilisation des caractères graphiques-souris que seule celle-ci est capable d'imprimer.
Deux types d'impression sont disponibles:

- pica standard en qualité brouillon,
- qualité supérieure avec caractères demi-hauteur.

Ces deux types sont sélectionnés par ⇧-S, ⇧-H ou Option-S, Option-H.
En outre il est possible de sélectionner la couleur d'impression dans GE.CONFIG. Cette option n'est

active qu'avec une ImageWriter II et un ruban couleur.

Enfin, et c'est l'une des plus grandes originalités de GE 80, on **peut créer toutes sortes de fenêtres avec ou sans cadre et les sauvegarder avec leur dimension.** La majorité des fonctions d'édition décrites plus haut sont disponibles en mode fenêtre. Les fenêtres peuvent être déplacées verticalement et horizontalement grâce aux commandes Ó -> et Ó <- pour rechercher leur position optimale sur l'écran. Il est possible d'opérer des superpositions de fenêtres en utilisant alternativement la commande Ó-R.

Un gestionnaire de disques : GE.DSK

C'est probablement la partie de GE 80 la plus intéressante car elle constitue la meilleure illustration des possibilités de stockage et d'affichage quasi-instantané d'écrans ou de fenêtres d'écran.
On accède aux fonctions de gestion 'disque' par ESC et l'on découvre un menu déroulant à barres comprenant toutes les commandes classiques attendues d'un interpréteur tel que BASIC.SYSTEM, y compris la possiblité d'éjecter en bloc ou sélectivement les disquettes 3"1/2, une fenêtre d'aide, une option quitter qui n'est ni plus ni moins qu'un sélecteur et lanceur de fichiers SYS et ce, en ne faisant appel qu'à Prodos et à la ROM moniteur.
Note technique : le lanceur est copié dans une zone réservée de GE.DSK qui le transfère à un emplacement paisible de la RAM ($1C00) pour dégager l'espace débutant en $2000 et permettre le chargement des fichiers SYS.
Il n'est nullement nécessaire que BASIC.SYSTEM soit actif. Une sous-option de 'quitter' permet un retour au basic, s'il est disponible, sinon on peut lancer BASIC.SYSTEM à condition qu'il soit dans le même catalogue ou sous-catalogue.
Le gestionnaire de disque permet également de changer de volume en affichant tous ceux qui sont en ligne, indique ceux qui ne le sont pas et affiche le tout dans une fenêtre (maximum 12 volumes, conformément aux possibilités de Prodos 8).
GE.DSK se présente sous forme de **menus déroulants multi-fenêtres** aux options et commandes totalement transparentes à l'utilisateur, avec une gestion rigoureuse des erreurs (affichage de messages en clair au lieu de mystérieux codes d'erreur MLI), ce qui lui donne une facture trés professionnelle et un 'look' conforme à ce que l'on peut exiger aujourd'hui en mode texte de nos Apple //.

Cette seule partie représente probablement une des principales qualités de GE 80 par un conséquent travail de programmation en assembleur et le souci de **permettre à travers le source DSK.S, la récupération d'une importante librairie de routines optimisées et structurées.** Sans aucune prétention, je recommande l'étude des sources, établis et documentés dans un souci de transparence. A titre d'information, GE 80 représente environ 8500 instructions et 150 pages de listing.

GE.DSK est réutilisable en bloc moyennant quelques modestes aménagements et adaptations pour l'affranchir de ses liens et d'appels à certaines routines de GE.EDIT qu'il conviendrait dans ce cas de rapatrier. **Il pourrait servir de 'disk manager' pour tout programme nécessitant une gestion d'entrées/sorties.**

GE.DSK propose les options suivantes:
- catalogue, sous-catalogues et information fichiers
- changer de volume
- gestion de la date (si pas de carte horloge pour 2e)
- affichage d'un écran
- sauvegarde d'un écran sur disque
- protéger, cacher un fichier
- renommer un fichier
- supprimer un fichier
- créer un sous-catalogue
- éjecter un ou plusieurs disques
- aide à la gestion disque
- quitter

A noter que les écrans ou fenêtres peuvent être sauvés soit en RAM par la commande Ó-W, soit sur disque, à partir du menu 'options'ou, au cours d'un travail, par Ó-S dans le style d'Appleworks, en proposant le nom du dernier écran chargé.
Note technique: les écrans ou fenêtres sont sauvegardés avec leurs coordonnées et sous forme de fichiers de type $F5 (SCR). Si, en sélectionnant l'option 'sauver' dans le menu général, on appuye sur Ó-↵, l'écran ou la fenêtre sera sauvegardé sans ses coordonnées et sa longueur correspondra strictement au nombre de caractères affichables à l'écran; le fichier, dans ce cas, sera de type TXT pour une récupération éventuelle dans un traitement de texte ou un éditeur quelconque.

Deux autres segments, dont l'un est indispensable, accompagnent GE.SYSTEM:

GE.TOOLS (l'indispensable utilitaire d'application)

GE.TOOLS est un petit fichier de 463 octets permettant, à partir d'un programme Applesoft, Pascal ou Assembleur, de charger les écrans ou fenêtres générés par GE 80, ceux-ci s'affichant instantanément et, en prime, lorsqu'il s'agit d'une fenêtre, GE.TOOLS permet de générer un menu déroulant s'inscrivant dans celle-ci. Pour une fenêtre de N lignes, la barre haute du menu est placée à la ligne 4 et la barre basse du menu est à la position N-3.
GE.TOOLS permet de sauver le fond de fenêtre et de le restituer à la fermeture de celle-ci.
Pour illustrer les possiblités de GE.TOOLS, il est fourni un petit programme Applesoft intitulé GE.DEMO qui utilise toutes les fonctions internes de GE.TOOLS (afficher un écran ou une fenêtre, sauver un fond de fenêtre ou un écran, générer un menu déroulant à partir d'une fenêtre donnée, récupérer le numéro de l'option sélectionnée par RETURN à l'adresse décimale 30 (par un PEEK (30) en basic applesoft).
La routine GE.TOOLS étant **relogeable**, celle-ci peut être placée n'importe où, par exemple entre Prodos et ses fichiers en allouant le nombre de pages mémoire nécessaires au stockage de la routine et des écrans ou fenêtres (voir annexe).

Il serait trés facile de transformer GE.TOOLS en commande externe en utilisant la routine CMDLOAD bien connue des 'Applemaniaques'. Autre formule: à la suite d'un programme basic.

GE.CONFIG (un 'plus' pour personnaliser GE.SYSTEM)

Il s'agit d'un petit fichier SYS écrit également en assembleur et en harmonie avec le look de GE 80, qui permet de **'patcher' GE.SYSTEM** pour fixer une présentation par défaut, au lancement, à savoir:

- cadre ou pas cadre,
- plein écran ou écran avec barre de commande,
- mise en place ou non d'une règle de tabulation,
- clavier sonore ou silencieux,
- insertion ou écrasement,
- choix de la couleur d'impression.

Ce fichier peut être lancé soit à partir de GE.SYTEM par le sélecteur interne, soit indépendamment. Il n'est opérant que si GE.SYSTEM est présent dans le même catalogue ou sous-catalogue, sinon il y a affichage d'un message d'erreur. Une fois la configuration effectuée, une option permet d'accéder directement à GE.SYSTEM.

Enfin divers fichiers 'écran' ou 'fenêtre', du type $F5 ou SCR comme (SCR)een, sont fournis à titre d'exemples. On notera la présence d'un fichier 'écran' nommé **GE.BITMAP** particulièrement intéressant pour les lecteurs qui souhaiteraient tirer parti des routines de GE 80. Cet écran se présente sous forme d'un tableau donnant toute la structure et l'emplacement des principaux segments de GE 80.

En conclusion, GE 80, c'est:

- GE.SYSTEM, fichier SYS qui comporte entre autres, deux segments implantés selon les adresses fournies dans GE.BITMAP:
- l'éditeur GE.EDIT et,
- le gestionnaire de disque GE.DSK
- GE.CONFIG, fichier SYS
- GE.TOOLS, fichier BIN
- GE.DEMO, fichier BAS.

GE.DEMO est associé aux fichiers DEMO.MASK, DEMO.MENU et DEMO.MSG

Tous les sources sont fournis et sont assemblés avec MERLIN. GE 80 est essentiellement bâti autour d'une routine simple et paramétrable (MEMO, située dans GE.DSK), de lecture/écriture d'écrans ou fenêtres-texte en 80 colonnes entre la RAM et la mémoire écran.

Cette routine est d'ailleurs utilisée constamment dans le développement de ce logiciel pour gérer toutes sortes de fenêtres à ouvrir ou fermer sans perdre d'information.

A propos de GE 80: appuyez sur Option-? quand vous êtes dans l'éditeur!

Exemple d'utilisation de GE.TOOLS dans un programme BASIC APPLESOFT

1. On réserve une zone mémoire en abaissant HIMEM et en allouant le nombre de pages-mémoire nécessaires pour stocker:
   - l'écran initial (masque ou fond d'écran),
   - la fenêtre de menu,
   - le fond de fenêtre,
2. On charge GE.TOOLS à l'adresse HIMEM,
3. On charge le masque et la fenêtre de menu,
4. On assigne à des variables les adresses respectives de HIMEM et des zones de stockage de DEMO.MASK, DEMO.MENU et le fond de fenêtre (N, A, B, C),
5. Un écran ou une fenêtre s'affiche par:
CALL N,A où N = HIMEM, A = adresse écran
6. Un menu déroulant s'affiche de la même façon mais avec 3 paramètres:
CALL N,B,C où N = HIMEM, B = adresse menu, C = adresse fond de fenêtre.

On dispose de la commande ESC pour quitter le menu et revenir à l'écran initial avec restauration du fond de fenêtre. On sélectionne une option du menu par ↵. Le numéro de l'option choisie est récupérable par PEEK (30).

On trouvera ci-après, le listing d'un petit programme intitulé GE.DEMO, illustrant la méthode précédente:

Listing de 'GE.DEMO'

```
10 rem ---------------- préparation ----------------
30 d$ = chr$(4)
40 print d$"pr£3":print
50 print d$"-ALLOCATE":rem -- alloue 17 pages-
   mémoire --
60 print d$"bload GE.TOOLS,TREL,A$8800"
70 print d$"bload DEMO.MASK,T$F5,A$8A00"
80 print d$"bload DEMO.MENU,T$F5,A$9184"
90 print d$"bload DEMO.MSG,T$F5,A$9488"
100 himem:34816
120 rem ---------------- programme ----------------
140 N = 34816:A = 35328:B = 37252:C = 38024
150 call N,A:rem --------- affichage du menu ---------
160 poke 49168,0:wait -16384,128:K = peek (49152)
170 if K <> 155 then 160
180 call N,A:rem------------ affichage du décor ---------
190 wait -16384,128:K=peek (49152)
200 if K = 155 then 160
210 if K = 141 then poke 49168,0
220 NB = peek (30)
230 call N,C:rem ------ affichage du message final -----
240 poke 1403,54:vtab 14:print NB
250 poke 49168,0:wait -16384,128
260 home:if NB <> 3 then 150:rem -- option 3 = sortie
270 poke 49168,0:end
```

NB: *ALLOCATE* est une routine très courte, implantée en $300, qui alloue le nombre de pages-mémoire nécessaires à ce programme (il est rappelé qu'une page-mémoire a une longueur de 256 octets). La routine en assembleur correspondante est la suivante:

```
ORG  $300          ; adresse d'assemblage
JSR  $BEF8         ; ré-initialise les buffers
LDA NB_PAGES       ; nombre de pages à allouer
JMP  $BEF5         ; allocation de NB_PAGES
```

Il est commode et plus simple d'introduire cette routine dans le programme basic précédent par des DATA (8 valeurs à lire) de la façon suivante:

```
50 for I = 0 to 7:read J:poke 768+I,J: next I
55 call 768
[.....]
300 data 32,248,190,169,17,44,245,190
```

Le nombre 17 (souligné) représente le nombre de pages à allouer.

– Eh, mais GE 80, c'est du 8 bits ! Sur un GS !
– Oui, comme InWords...

GE 80 : Principales commandes de l'éditeur d'écrans et fenêtres

Touche	Fonction	Touche	Fonction
→	Avance d'un caractère	←	Recule d'un caractère
↑	Monte d'une ligne	↑	Descend d'une ligne
⌘ ←	Curseur en début de ligne	⌘ →	Curseur en fin de ligne
⌘-↑	Curseur en haut de l'écran	⌘-↓	Curseur en bas de l'écran
Option-↑	Curseur en début d'écran	Option-↓	Curseur en fin d'écran
Option →	Avance d'un mot	Option ←	Recule d'un mot
TAB	Avance d'une tabulation	⌘-TAB	Recule d'une tabulation
DEL	Efface 1 car. à gauche	⌘-DEL	Efface 1 car. sous le curseur
⌘-G	Efface à gauche du curseur	⌘-D	Efface à droite du curseur
⌘-ESC	Efface l'écran/video normale	Option-ESC	Efface l'écran/vidéo inverse
⌘-M	Insère une ligne	⌘-O	Supprime une ligne
⌘-L	Copie une ligne	⌘-C	Copie 1 chaîne de caractères
⌘-W	Sauve l'écran en MEV	⌘-R	Affiche 1 écran stocké en MEV
⌘-S	Sauvegarde l'écran sur disque	Option-W	Duplique 1 écran en MEV
⌘-M	Mise en mode fenêtre	Option-M	Edition d'une fenêtre
Option-?	A propos de GE.80	⌘-Q	Quitter GE.80

Touche	Fonction ou effet	Observations
⌘-X	Mode NOR/INV/MOUSE	>
⌘-A	Insertion/recouvrement	> Commandes
⌘-V	Ecrit vertical/horizontal	> de type
Option-S	Avec ou sans son	> 'bascule'
Option-F	Avec ou sans ligne de commandes	>
Option-C	Avec ou sans cadre	>
Option-T	Avec ou sans réglette	>
Option-I	Conversion Inv. <-> nor.	> Mode normal
Option-M	Conversion Maj. <-> min.	> tapez: ↵

⌘-↵ ESC	Mode fenêtre / mode normal	> Déplacer le curseur
Option-↵ ESC	Edite une fenêtre / retour mode normal	> vers l'extrémité
⌘ → ⌘ ←	Déplace une fenêtre (droite / gauche)	> de la diagonale
⌘-↑ ⌘-↓	- d° - (haut / bas)	> et ↵
Option-F Option-C	Pleine fenêtre / avec ou sans cadre	>

« C'est pas grave : moi, je les ai déjà... »

Mais vous, il faudra que vous attendiez les prochains ToolBox Mag. Ils sont déjà dans nos tiroirs: restez branchés, et vous aurez (entre autres):

- Un NDA "Lynx" de Patrick Desnoues (?).
- Une étude de Jacques Liautard "Il ne lui manque même pas la parole", ou comment faire parler le GS...
- L'anatomie des disquettes GS/OS 5.04, suite du "GS pour débutants" d'Eric Weyland.
- Une revue logicielle de l'indispensable Genesys 1.2, par un expert: l'ami François Uhrich.
- Un (dé)-compilateur de ressources complet de Jacques Destelle: adieu à la galère Rez !
- Une étude de votre serviteur sur la Personal LaserWriter :"La Laser sans le Mac..."
- Un excellent lanceur de Jacques Destelle, qui rendra Wings largement superflu...
- Une étude de notre ressourcier Bernard Fournier sur la cohabitation, et même la collaboration, entre le Pascal et l'Assembleur.
- Une série d'études du productif Jacques Destelle sur le passage simultané du 8 au 16 bits et de l'Applesoft au Pascal.
- And much, much more...

Ne vous étonnez pas si j'ai du mal à croire que l'Apple II GS est mort... J.Y. Bourdin

Les Mégas sans la galère

Notes sur la gestion des disques durs

J.Y. Bourdin

Les mégas à la pelle, c'est bien beau, et c'est indispensable aujourd'hui pour qui veut réellement utiliser son GS. Mais cela pose des problèmes nouveaux, toujours nos "problèmes de riches": il faut ranger et gérer tout ça. Cet article voudrait montrer aux utilisateurs de disque dur comment se sortir d'une des galères les plus fréquentes (à l'optimisation), et aux utilisateurs de lecteurs de cartouches Syquest comment en finir avec la galère.

"Ça plante quand j'optimise"

Vous avez commencé à optimiser votre disque dur, c'est-à-dire en fait à le ranger. Et voilà qu'au beau milieu, tout plante. Du coup, en guise de rangement, vous avez un bazar terrible. Et vous commencez à injurier Prosel ou Vitesse: parce qu'en plus, ce n'est pas la première fois! Alors, pourquoi, et que faire?

Trois raisons possibles au plantage, dont deux inévitables:

1/ **Un plantage soft du logiciel ou du système.** Personne n'est parfait, ce genre d'accident n'est pas évitable par principe: l'optimisation d'un disque dur demande énormément de travail très complexe au système. Ce système peut par ailleurs être assez fragile: le driver de disque dur SCSI d'Apple est encore loin d'être parfait , même chose pour la carte SCSI-DMA Apple. La vitesse a un prix.

2/ **L'apparition de "bad blocks" en plein milieu du travail.** Là non plus, ce genre de chose n'est pas évitable, par principe. Voir ci-dessous.

3/ Quelque chose que vous pouvez éviter: **le manque de mémoire dans votre GS, ou le manque d'espace libre sur le dur.** Quand vous faîtes ce travail, chassez les accessoires, détruisez quelques jeux ou archives inutiles sur le dur. **Un disque dur ne devrait jamais être rempli à plus de 80% de la capacité de chaque partition.**

L'expérience m'enseigne que la raison la plus fréquente est en fait la seconde: l'apparition de bad blocks. Même quand vous avez fixé les bad blocks auparavant avec Prosel ou Delivrance, il s'en crée au beau milieu de l'optimisation! **C'est vicieux, c'est diabolique, mais... c'est normal!**

L'opération délicate, sur un disque, n'est en effet pas tant la lecture que l'écriture. **C'est au cours des opérations d'écriture que les bad blocks se fabriquent.** Or, il y a des zones de votre disque dur sur lesquelles vous n'écrivez jamais: celle où il y a les 780k du fichier Appleworks-GS, par exemple, est souvent lue, mais jamais écrite. **Sauf** dans un cas: quand vous optimisez le dur, précisément.

Le programme optimiseur aura donc lu un bloc, puis l'aura écrit, puis deviendra incapable de relire le même bloc: nul ne peut se tirer de ce genre de situation. Joint aux pépins soft eux aussi inévitables, une conclusion s'impose: **l'optimisation du dur est par nature une opération dangereuse!** Les documentations le disent, mais il faut l'expérience pour le comprendre: **"C'est dangereux" signifie "Ça va planter"** !

La première règle est donc la **prudence**: avant toute optimisation, faire un back-up, bien sûr. Mais pas seulement: vérifier l'intégrité des catalogues, emprisonner les bad blocks (rappelons que les bad blocks sont chose parfaitement normale sur un disque dur).

De ce point de vue, le programme **"Deliverance", de Vitesse**, a une option très intéressante: **le réglage de la sensibilité dans la lecture des bad blocks**. Si on fait confiance au système, celui-ci va relire plusieurs fois (jusqu'à 30 fois pour GS/OS) un bloc qu'il n'arrive pas à lire. Il va donc accepter des blocs douteux. J'ai 6 bad blocks sur la partition de 43000 blocs sur laquelle je travaille en ce moment: je me moque parfaitement d'en avoir 100 ou 200, si cela m'assure que tous les blocs douteux sont mis à part, donc que je ne tomberai pas inopinément sur un bad block.

Cela dit, il y aura toujours apparition de nouveaux bad blocks à l'optimisation: si le programme de vérification des bad blocks devait

faire autre chose que lire, par exemple lire et écrire 30 fois chaque bloc, il faudrait 24 heures pour vérifier un dur, et cela n'empêcherait pas l'apparition de quelques nouveaux bad blocks à la 31e fois. Et il y aura toujours le risque de plantages soft.

La conclusion de tout ça, c'est que la qualité principale d'un programme d'optimisation du dur n'est pas sa rapidité, ni sa beauté, ni toute autre considération subalterne. Ce n'est pas non plus qu'il ne plante pas: le programme ou le système planteront forcément une fois ou l'autre.

En fait, **la qualité principale d'un programme d'optimisation du dur, c'est sa capacité à laisser les choses les plus propres possibles en cas de pépin**. De ce point de vue, je dois dire du mal de **Renaissance, de Vitesse**: il m'a laissé deux fois un disque dur bon à reformater, avec rien, absolument rien, de récupérable!

Dans l'état actuel de mes connaissances, je ne peux dire du bien que de deux programmes optimiseurs: **Beach Comber version 3.7 de Prosel 8** (eh oui, il optimise parfaitement les fichiers étendus de GS/OS, même sous P8), assez lent mais très fiable, **et Prosel 16, dans ses versions récentes (8.55 et au-delà)**. Passant par GS/OS et son driver SCSI, il est plus rapide que Beach Comber 8, mais moins fiable. Seulement, quand l'optimiseur de Prosel 16 plante, il laisse au maximum deux ou trois fichiers abîmés. Repassez au vérificateur de volumes, détruisez ces deux ou trois fichiers, vérifiez le catalogue, repassez au chasseur de bad blocks, recopiez les fichiers détruits à partir du backup, et... vous pouvez relancer une nouvelle optimisation!

Il m'est arrivé fréquemment de m'y reprendre à deux ou trois fois pour optimiser un dur avec Prosel. Il laisse les choses suffisamment propres pour qu'on puisse reprendre le travail ensuite. Autre supériorité de Prosel: sa fonction d'optimisation dite "Turbo", qui se contente de défragmenter les nouveaux fichiers qui ont été fragmentés depuis la dernière optimisation, sans reprendre le rangement d'ensemble du dur.

Bref, optimiser, c'est dangereux. Soyez prudents, et comme d'habitude: "Prosel, ou tu meurs".

Un dernier conseil: avec un système et des programmes qui utilisent en permanence le relais sur disque, il est préférable de ne pas demander à ces systèmes et programmes de s'optimiser eux-mêmes en plein travail! Quand vous peignez votre dur, il est bien préférable de le faire à partir d'**un mini-système avec Prosel que vous aurez booté à partir d'une disquette**. Si ça plante, vous pourrez toujours repartir de la disquette.

Patch Chinook

Voici un patch pour les utilisateurs des Chinook Utilities version 1.5, pour avoir des " I " à la place des "ù":

```
§ BLOAD CSU.1.5.SYSTEM, A$2000,T$FF
§ CALL - 151
* 6826: A1
* BSAVE CSU.1.5.SYSTEM, A$2000, T$FF
```

JYB

Syquest: la galère des back-ups, c'est fini!

N'hésitons pas à l'écrire, **il y a quelque chose de mieux qu'un Syquest**, en matière de mémoire de masse pour le GS: **deux Syquest**, bien sûr! Depuis que j'ai le second, je me demande comment j'ai pu vivre avec un seul lecteur: puisque c'est un lecteur de disquettes, il en faut deux, pour les copies. Ne faîtes plus de back-ups sur disquettes 3,5, copiez tout simplement. N'optimisez plus la cartouche après l'avoir restaurée depuis le back-up: votre copie est déjà optimisée. **Le second Syquest, c'est la fin de la galère des back-ups.**

Attendez, ne partez pas! Ce que vous ne savez pas, c'est que **le second Syquest est gratuit**: vous l'avez déjà si vous avez le premier.

Comment se fait-il qu'il m'ait fallu presque un an pour m'en rendre compte, alors que c'est totalement évident? Et comment se fait-il que personne à ma connaissance ne s'en soit rendu compte avant? Eh bien, j'avais gardé mes habitudes du disque dur avec back-up sur disquettes, voilà. Le poids de l'habitude, nous en mourrons tous...

Voici donc ce truc évident: votre cartouche Syquest fait 42 Mégas, et vous en avez bien entendu une autre pour les back-ups. Tant que nous gardons la limite de 32 Mégas du FST Prodos de GS/OS, cela signifie qu'il faut faire deux partitions par cartouche, soit 4 partitions en tout. Le truc, c'est tout simplement de partitionner chaque cartouche de 42 Mégas en deux partitions

strictement égales en nombre de blocs (43 008 blocs, 21 504k), et de **consacrer la deuxième partition de chaque cartouche au back-up de la première.**

Le back-up d'une partition de 21 Mégas se fait **en sept minutes trente** montre en main sur ma configuration. Et pendant ces sept minutes, je vais boire tranquille un café, sans toucher à une seule disquette ni m'occuper de rien. Il suffit de copier la première partition sur la seconde, et le back-up est fait!

Bien entendu, pas une copie fichier par fichier, et surtout pas avec le Finder: non seulement ce serait beaucoup trop long, mais il faudrait ensuite optimiser la seconde partition, comme la première.

Il faut faire une **copie physique, par blocs**: c'est pour cela que vous avez créé deux partitions de 21 Mégas par cartouche, strictement identiques en nombre de blocs. Comme toujours, n'essayez pas de faire ça avec le Finder: ça devrait marcher (en faisant glisser l'icône de la première partition sur la seconde), mais ça ne marche pas, bien entendu (erreur $FE04, pourquoi pas celle-là). En revanche, évidemment, ça marche comme sur des roulettes avec la commande **"Copy Volume" de Prosel 16**! Prosel, ou tu meurs...

Dès que la copie est terminée, renommez la seconde partition pour qu'elle ne garde pas le même nom que le première, et c'est fini. Au lieu d'avoir une cartouche "originale" de 42 Mégas et son back-up sur une autre cartouche de 42 Mégas, vous avez **deux cartouches "originales" de 21 Mégas contenant chacune son propre back-up!**

Léger inconvénient de cette formule: toutes vos partitions comprendront une copie du système, alors qu'avec la formule habituelle, la seconde partition n'en avait pas besoin. Vous perdez donc un peu de place. Je ne compte pas comme un véritable inconvénient la nécessité de changer de cartouche pour passer à la partition avec Xénocide: expérience faite, ce n'est pas gênant de n'avoir que 21 Mégas en ligne en même temps, quant il suffit de changer de cartouche.

Et la sécurité ?

Du point de vue de la sécurité, malgré les apparences, cette formule est très sûre: si votre Syquest tombe en panne, vos data sont toujours là sur la cartouche, et vous pouvez les récupérer sur le Syquest du copain. Si c'est la cartouche qui tombe en panne, cela ne se fait que d'une seule façon à ma connaissance: par l'apparition de bad blocks. Or, il est statistiquement impossible que ces bad blocks apparaissent sur les mêmes blocs des mêmes fichiers sur les deux partitions en même temps: il y aura toujours, pour chaque fichier, une des deux partitions de bonne.

Même si les deux blocs 2 sont abîmés ensemble (!), effacez les partitions sans les reformater (commande *Erase* de Prosel 16 et GS/OS), et vous récupérerez tout ce qui est en sous-catalogue (toujours avec Prosel 16). Si GS/OS ne reconnaît même pas deux partitions à effacer, rien n'est encore perdu: repartitionnez allègrement la cartouche (sans reformater physiquement, bien entendu), en reconstituant deux partitions strictement égales, puis faîtes comme ci-dessus (effacer et récupérer le contenu des sous-catalogues). Pour partitionner sans formater, les utilitaires disques du système Apple ne conviennent pas plus que le Finder ne convient pour copier les disques: j'utilise les **Utilitaires Chinook version 1.5** (attention, ces utilitaires ne marchent qu'avec une carte SCSI Apple).

Je ne vois de vrai danger à cette solution que dans un seul cas: la carte SCSI se réserve au début de votre cartouche 32 blocs pour une partition "invisible" appelée "Apple_Partition_Map". Si le premier de ces blocs invisibles, qui est aussi le premier bloc physique SCSI de la cartouche, est devenu un bad block, essayez de repartitionner comme indiqué ci-dessus: généralement, on peut réécrire ce bloc. Si ce n'est pas possible, il y a encore une solution: demandez aux utilitaires Chinook de réassigner les bad blocks (c'est une commande SCSI), puis repartitionnez. Neuf fois sur dix, ça marche. Mais la dixième fois, votre cartouche est irrécupérable (pas absolument d'ailleurs, mais nous entrons alors dans le domaine de la bidouille SCSI impubliable), et vos data sont perdues.

Il y a donc une chance sur 86 000 pour que vous ayez un bad block en bloc physique zéro, et dans ce cas, une chance sur dix pour le désastre. **Une chance sur 860 000 d'aller au désastre**, tel est le prix du "second Syquest gratuit". Je ne connais pas encore d'exemple où ce désastre du bloc 0 illisible soit arrivé, mais 1/860 000, **ça n'est pas zéro**. Si vous refusez ce prix, ou si vous voulez aussi vous protéger contre les risques de destruction physique de la cartouche (tasse de café, B52, etc) vous pouvez continuer avec les back-ups sur disquette 3,5, ou... vous offrir un second Syquest.

Récupérer un source Complete (TML) Pascal sous Orca Pascal

Michel Racine

L'éditeur "Prizm" (voir article de François Uhrich sur Orca C dans le n°1) est très performant mais assez susceptible: vous ne pourrez charger le fichier source (de type texte) que si la dernière ligne est vide (retour chariot seul). Un passage à travers l'éditeur d'Orca en mode texte (Entrez successivement "Edit MonTexte", "Control Q", "S"), résoud le problème.

Lors du chargement dans Prizm ou dans l'éditeur en mode texte, le 7ème bit de tous les caractères de code > $7F (nos chers accents) est mis à zéro.

Pour gérer ces caractères, définir des variables: *var eAigu: char;* les initialiser: *eAigu := chr($8E);* puis remplacer des mots comme *'insérez...'* par *'ins', eAigu, 'rez...'*

Les versions actuelles des éditeurs ne gèrent pas les caractères de tabulation (Control I) qui apparaissent en vidéo inverse, ainsi que tous les caractères accentués dont le 7ème bit a été mis à zéro. Il est possible de remplacer tous ces caractères par des espaces en faisant passer l'ensemble du texte par le presse-papier, ou en utilisant l'utilitaire DETAB pour les tabulations.

Pour un source très simple comme celui de l'unité ST le travail est presque terminé: il suffit d'ajouter des parenthèses aux paramètres qui suivent le mot clé "Tool" et de modifier quelques identificateurs: *MaxInt* remplacé par *32767* et *string* par *pString*.

Mais souvent il reste encore beaucoup à faire. Il faut comparer les directives de compilation, les identificateurs d'unités, les identificateurs des types définis dans ces unités, ainsi que ceux des procédures et fonctions (et je ne suis pas exhaustif). L'option *"Check for Errors"* du menu *"Run"* de Prizm aide à repérer les modifications nécessaires.

La commande *"Canon"* d'APW, utilisée avec le dictionnaire adéquat, devrait pouvoir automatiser les remplacements. Possédant seulement la documentation d'Orca Pascal il m'est difficile d'établir ce dictionnaire. En particulier, il faut comparer les identificateurs de toutes les fonctions et procédures effectuant les appels à la Toolbox ainsi que ceux des structures de données liées à la Toolbox (et donc posséder le source des interfaces Toolbox pour les deux compilateurs).

Pour les programmes utilisant les ressources, conservez soigneusement la portion ressources du programme compilé ou les fichiers de type $00 dont le nom se termine par ".r". Modifier les ressources sous Orca suppose d'avoir installé *DeRez* et *Rez*.

Pour terminer, déroulez le menu *"Languages"* de Prizm et choisissez *"Pascal"* avant de sauvegarder (ou utilisez la commande *"CHANGE MonTexte Pascal"* d'Orca).

Tool 219 : version 1.1.1

Dans le sous-catalogue /TOOL219.V.1.1.1, vous trouverez la dernière version de notre SoundSmith Tool. Juste des corrections de bugs, dont un sérieux.

Faute de place sur la disquette, nous n'avons mis qu'un fichier-source de compléments et de corrections au source Merlin publié dans ToolBox Mag 3. A vous de faire le copier-coller nécessaire.

Dans /TOOL219.V.1.1.1/ORCA.PASCAL, vous trouverez les interfaces du Tool219 pour Orca Pascal par Michel Racine. Attention, le fichier *STLib* doit venir impérativement *avant PasLib* dans votre catalogue /LIBRARIES.

Michel Racine en a profité pour nous envoyer quelques notes sur la traduction d'un source TML en source Orca Pascal (voir ci-contre): ce n'est pas si facile...

Le Tool219 va bientôt être le Tool le plus documenté et le mieux débugué du GS. En tout cas, c'est vraiment une œuvre collective... JYB

Premiers conseils sur HyperCard

Donc, les piles HyperCard, c'est parti mon kiki, en musique et en couleurs. Bien entendu, les prochains ToolBox Mag vont vous présenter HyperCard, et vous permettre de tirer le maximum de ce qui est bien **l'Applesoft des années 90.**

Mais en attendant, quelques conseils tout de suite pour ne pas être perdu dans HC:

• D'abord, rappelez-vous que c'est HyperCard: on ne sauve pas, **tout est sauvé sur disque en temps réel**. Une fausse manœuvre dans votre pile "Home" est enregistrée immédiatement! Les back-ups sont une **nécessité absolue** avec HC.

• Beaucoup de piles cherchent à vous faire naviguer "bêtement", "en aveugle", dans HyperCard, comme sur un vulgaire Mac. Pour rester sur GS, et naviguer sans crainte dans HC, il y a **trois commandes à garder en mémoire**, et trois seulement:

• **Ó-Espace**: donne **accès à la barre de menus** d'HyperCard qu'on voulait vous cacher.

• Pour **avoir accès à tous les scripts** qu'on vous dissimule, par exemple ceux des boutons: **Ó-Option-Shift**, puis un **double-clic** sur la zone dont on veut étudier le script.

• Ó en déroulant le **Menu File** vous donne l'accès à l'item du menu qui permet de **déprotéger les piles soit-disant "protégées".**

JYB

Les Ressources du GS, troisième Partie :

Modifier ses ressources "à la volée"

Bernard Fournier

Dans nos deux premiers articles, nous avons étudié comment étaient constituées les ressources. Nous n'avions alors abordé le problème que sous l'aspect de la création/modification de ressources avec des éditeurs de ressources.
A l'occasion des exemples, nous avions vu la souplesse apportée par les ressources qui permettent de stocker tous les fichiers de données d'une application. Partant de là, on peut fort bien imaginer de conserver sous forme de ressources les meilleurs scores d'un jeu, ou bien la configuration par défaut de tel ou tel logiciel.
A priori, rien ne s'y oppose, bien au contraire direz-vous... Eh bien non ! En informatique comme dans bien d'autres domaines, il ne faut pas vendre la peau de l'ours trop tôt! Et dans le cas présent: la modification de ressources par voie logicielle est... disons *pas évidente du tout*. Apple y a d'ailleurs consacré une Note Technique (83).

NB: nous suivrons ici grosso modo les conseils d'Apple, tout en sachant que cela laisse subsister un problème (double chargement des ressources marquées PreLoad). Eventuellement, nous pourrons étudier dans un prochain numéro comment contourner ce problème.

Le problème

Pour constater le problème, lançons l'application UPDATEFALSE qui se trouve sur la disquetteToolBox Mag 4. Que se passe-t-il ? Deux ressources *rPstring* sont affichées successivement et on tente de les modifier par *UpDateResource* puis par *WriteResource*: dans les deux cas l'erreur $4E est retournée et nos ressources ne sont pas modifiées sur disque.
Que signifie ce code d'erreur ? C'est un message GS/OS signifiant que le code d'accès au fichier interdit l'opération en cours. Intéressant. Cela veut donc dire que le fichier ressource de l'application en cours est protégé contre l'écriture. Le remède est évident: il suffit de trouver le moyen de déprotéger le fichier ressource de l'application courante pour que tous nos malheurs soient résolus.

Les causes

Réfléchissons un peu. Si on ouvre un fichier ressource quelconque et que l'on modifie une de ses ressources, celle-ci est sauvée sans problème sur disque (c'est ainsi que l'on peut modifer des ressources avec Genesys par exemple). Cependant un problème survient lorsqu'on veut qu'une application *modifie elle-même ses ressources*. Quelle peut bien être la cause de ce phénomène ?
Tout simplement lorsqu'on édite un fichier ressource quelconque, on procède comme suit:

- *OpenResourceFile*
- *LoadResource*
- *MarkResourceChange*
- *UpDateResource* ou *WriteResource*
- *CloseResourceFile*

Et lorsqu'on travaille sur le fichier ressource de l'application en cours, on fait:

- *LoadResource*
- *MarkResourceChange*
- *UpDateResource* ou *WriteResource*

Pas besoin de chercher plus loin: dans un cas on procède manuellement à l'ouverture/fermeture du fichier ressource, et dans l'autres on laisse ce soin au Resource Manager par l'intermédiaire de *StartUpTools*. C'est ici la cause de nos tracas: *le Resource Manager ouvre le fichier ressource de l'application en cours en n'autorisant que la lecture*, d'où le code $4E lorsqu'on tente une modification.

Le remède

La cause étant trouvée, reste à trouver un moyen de 'feinter' le Resource Manager. De manière intuitive, on sait qu'il va falloir ouvrir et fermer nous-mêmes les ressources, ceci afin d'avoir l'autorisation d'écriture sur disque.
Cependant, il ne faut pas oublier que le Resource Manager ouvre systématiquement les ressources de l'application lors du startup !
Donc marche à suivre:

- fermer le fichier ressource de l'application courante
- ouvrir manuellement le fichier ressource de l'application courante
- faire les modifications de notre choix
- fermer manuellement le fichier ressource de l'application courante

Regardons les paramètres à passer à ces instructions:

- *CloseResourceFile:* admet un *FileId* comme argument. Cet identifiant sera fourni par *GetCurResourceFile*, ce qui permettra de fermer le fichier ressource de l'application en cours.
- *OpenResourceFile:* il faut lui passer un pointeur sur le nom du fichier ressource à ouvrir. C'est à dire concrètement un pointeur sur le nom de l'application en cours. Pour récupérer le nom de l'application en cours, on peut employer *GetNameGS*.
- Enfin la deuxième fermeture manuelle se fera avec le *FileID* fourni par *OpenResourceFile*.

Alors allons-y:

```
pblockPtr.pCount:=1;
GetNameGS(pBlockPtr);
avec pBlockPtr: GetNameRecGs;
TempPtr:=pBlockPtr.dataBuffer;
```

```
avec TempPtr:ResultBuf255Ptr;
GS2PString(TempPtr^.bufString,TempStr);
CloseResourceFile(GetCurResourceFile);
FileId:=OpenResourceFile(0,nil,@TempPtr^.bufString);
```

On lance.... et surprise: ça ne marche pas ! Enfin plus concrètement, ça marche des fois oui, des fois non, mais plus souvent non.
En examinant de plus près l'exécution de ce programme, je me suis apperçu que l'erreur provenait de *GetNameGS*: en effet parfois le code $53 est retourné (paramètres invalides).
Pas de panique, analysons. *GetNameGS* admet un pointeur comme paramètre; ce pointeur désigne une structure constituée comme suit:

- un *pcount* (égal à 1)
- un pointeur sur une chaîne GSOS.

Et la chaîne GS/OS est elle-même constituée par:

- la longueur de chaîne (deux octets)
- la chaîne proprement dite.

Avec le programme que nous avons écrit, voici ce qui se passe:

- l'emploi de *pBlockPtr* dans *GetNameGS* fait bien empiler un pointeur
- ce pointeur référence une structure dont les deux premiers octets contiennent la valeur 1 (*pcount*) et les 4 suivants contiennent... 00 00 00 00 ! Ce qui fait que la chaîne retournée par la fonction (donc le nom de l'application) sera placée à l'endroit indiqué par le pointeur (donc en 00/0000 soit en plein dans la page zéro !).

Pas étonnant que ça coince ! On est en train d'écrire gaillardement sur des zones réservées.

Il faut donc utiliser une autre méthode: *déclarer préalablement des pointeurs pour chacune des structures employées et s'assurer que les zones pointées sont vides.* Pour cela, la fonction *NEW* est parfaite (à condition de déclarer correctement les variables afin de ne pas avoir de conflits de types ! Il y a des jours où on envie les programmeurs en assembleur...)

```
new(pBlockPtr);     avec pBlockPtr: ^GetNameRecGs;
new(pBufPtr);       avec pBufPtr : ResultBuf255Ptr;
```

ceci nous définit deux zones réservées pointées par *pBlockPtr* et par *pBufPtr*. Il reste maintenant à dire que le champ *databuffer* de *pblockPtr* pointe sur *pbufPtr*:

```
pBlockPtr^.pCount:=1;
pBlockPtr^.dataBuffer:=pointer(pBufPtr);
```

et on peut tranquillement faire notre appel

```
GetNameGS(pBlockPtr^);
```

et récupérer le nom de l'application dans une chaîne pascal *TempStr*:

```
GS2PString(pBlockPtr^.dataBuffer^.bufString,
TempStr);
```

Notez les quelques lignes suivantes affichant les valeurs des pointeurs: ces routines sont ici uniquement à titre pédagogique et montrent ***comment débugguer un programme Pascal sans outil spécial:***

```
i:=LongInt(@pBlockPtr);     on déréférence le pointeur
MoveTo(100,40);
theString:='    ';          on initialise la chaîne
Long2Hex(i,pointer(ord4(@theString)+1),6);
on transfère un octet après le début de la pString !!!
DrawString(concat('pBlockPtr : $',theString));
```

Bien, maintenant que l'on a récupéré le nom de l'application en cours, on peut procéder à l'ouverture manuelle de ses ressources:

```
FileID:=OpenResourceFile(0,nil,@pBlockPtr
^.dataBuffer^.bufString);
```

Ensuite, on charge une ressource:

```
MyHandle:=LoadResource(kResType,kResId1);
```

comme il s'agit de *pString* que l'on veut modifier, on va réserver la taille maxi:

```
SetHandleSize(256,MyHandle);
```

En effet, le MemoryManager et le Resource Manager vont charger la chaîne de 10 octets, par exemple, dans une zone libre de 10 octets. Il y a donc de fortes chances pour qu'il y ait d'autres données à la suite de cette chaîne, en mémoire. Lorsqu'on va modifier la chaîne (par l'exemple l'agrandir à 30 octets): si on n'a pas réservé l'espace suffisant pour cette modification, on va encore écrire allègrement sur des zones réservées... et bonjour les dégâts !
Pour les mêmes raisons, on va indiquer la taille réelle du Handle une fois la chaîne modifiée et prête à être sauvée (notez le +1 pour ajouter l'octet de longueur de la *pString*)

```
SetHandleSize(length(MyPointer^)+1,MyHandle);
```

Et enfin on indique au Resource Manager que cette ressource a été mise à jour:

```
MarkResourceChange(TRUE,kResType,kResID1);
UpdateResourceFile(GetCurResourceFile);
```

Ouf ! ***ça y est: nous avons réussi à modifier la ressource de l'application courante par voie logicielle.***

La suite du listing indique la même modification, mais à l'aide de *WriteResource*.

Ne pas oublier de laisser les lieux propres en quittant: on ferme le fichier ressource, on purge les pointeurs (*Dispose*) et les handles (*DisposeAll*) et on ferme les outils.

Moralité

La programmation en Pascal, pour aisée qu'elle soit, requiert une ***grande rigueur dans les déclarations.*** Lorsqu'on a à faire à des situations délicates, il suffit de regarder ce qui se passe en mémoire pour comprendre l'origine de l'erreur.

A cet effet, des outils comme *Debug Init* ou *Nifty List* sont d'un secours immense; mais à défaut on peut utiliser une routine d'affichage des pointeurs telle que celle décrite ici même.

Voila, j'espère que cet article vous aura apporté quelques solutions à vos problèmes, qui, si j'en juge mon courrier, avaient l'air d'être insolubles pour plus d'un et d'une. Comme vous l'avez remarqué, le Pascal complique un peu les choses, mais on y arrive.

Dans le prochain numéro, nous poursuivrons l'étude des ressources en abordant cette fois la définition de nos propres types.

Courrier des lecteurs

***A.Roybier:** Je rencontre des difficultés à charger les images et à lancer les programmes des disquettes ToolBox Mag. J'utilise GS/OS 4.0, TML Pascal, GS Paint, Deluxe Paint II.*

ToolBox Mag : Votre système est périmé. Nous en étions depuis un bon moment au 5.02 français, nous en sommes maintenant au ***5.04***. Avec le 5.04, vous pourrez lancer *PicMaster GS* de la disquette ToolBox Mag 2, qui vous permettra de convertir les images GS de toutes sortes dans des formats compatibles avec vos applications graphiques anciennes. Mais nous recommandons d'utiliser un programme graphique à jour, comme *Platinum Paint*. (voir ToolBox Mag 3).
TML Pascal en est aussi depuis un bon moment à la version II (TML Pascal II ou Complete Pascal II), et cette version est nécessaire pour éditer et compiler les sources TML Pascal de ToolBox Mag.

***Françoise Lau:** Je cherche depuis longtemps un programme graphique Desktop, un programme d'analyses statistiques genre Stat View sur GS, avec statistiques à 1 et 2 variables, moyenne, écart-type, variance et covariance, tracé de régression linéaire, d'histogrammes, etc. Avec bien sûr stockage des données dans un fichier sur disque, procédure de cumulation successive de données.*

ToolBox Mag : Nous ne connaissons effectivement pas d'application commerciale de ce type. Nous pensons que les programmeurs GS, tout particulièrement ceux qui programment en Pascal, devraient être intéressés par cette idée.

***Joël Desnoues:** A propos de Crystal Quest. Il y a 99 niveaux de jeu en tout (!). J'ai découvert l'emplacement où sont stockées les principales informations du jeu. Après avoir installé "Visit Monitor" et lancé Crystal Quest, passez sous moniteur et listez en $00/0A30:*

0A30:	*00 00 00*	*Total des points*
0A40:	*00*	*Niveau de jeu*
0A44:	*00 00*	*Nombre de vies*
0A46:	*00 00*	*Nombre de bombes*
0A48:	*00 00*	*Minutes*
0A4A:	*00 00*	*Secondes.*

Toutes ces valeurs sont en décimal, avec l'octet de poids faible en premier. Pokez les valeurs souhaitées aux bons emplacements pour découvrir tous les niveaux et éviter tout risque de foulure du poignet à Hubert.

ToolBox Mag : Votre truc ne produit pas exactement l'effet prévu: en tentant de battre nos scores faramineux, Hubert a frôlé de près la foulure...

***R.Burger:** Vos lecteurs ne sont pas forcément aussi à l'aise que vous avec les Tools: s'il vous plaît, n'ayez pas peur d'être explicites dans les commentaires des listings. D'autre part, il est assez désagréable de se trouver devant un source inexploitable parce qu'on n'a pas le langage adéquat. ToolBox Mag ne pourrait-il pas adopter un langage unique?*

ToolBox Mag : Vos demandes sont légitimes, et vous noterez, dans ce numéro même, notre effort pour les satisfaire.
Pour la seconde demande, plutôt que d'appauvrir le GS (la pluralité des langages est une richesse, nous attendons vos sources Forth ou Micol Basic), et de brider la liberté des programmeurs (pas de galère sur GS !), nous nous aiguillons vers la *lisibilité*, voire la *traductibilité* des programmes-sources. Nous espérons que la formule du plan, du synoptique du programme (ou même du projet de programme) sera reprise par de nombreux auteurs.
Par ailleurs, la formule d'avenir pour la programmation du GS est bien dans l'usage de *plusieurs langages ensemble dans le même programme*, un Linker se chargeant de coller les morceaux.
Si vous cherchez le meilleur investissement pour programmer le GS, nous vous recommandons donc:

- *Genesys*, de SSSI, pour les ressources, et :
- L'environnement de développement intégré *Orca* (ByteWorks), dans lequel vous pourrez intégrer tous les langages, en commençant par celui que vous préférez.